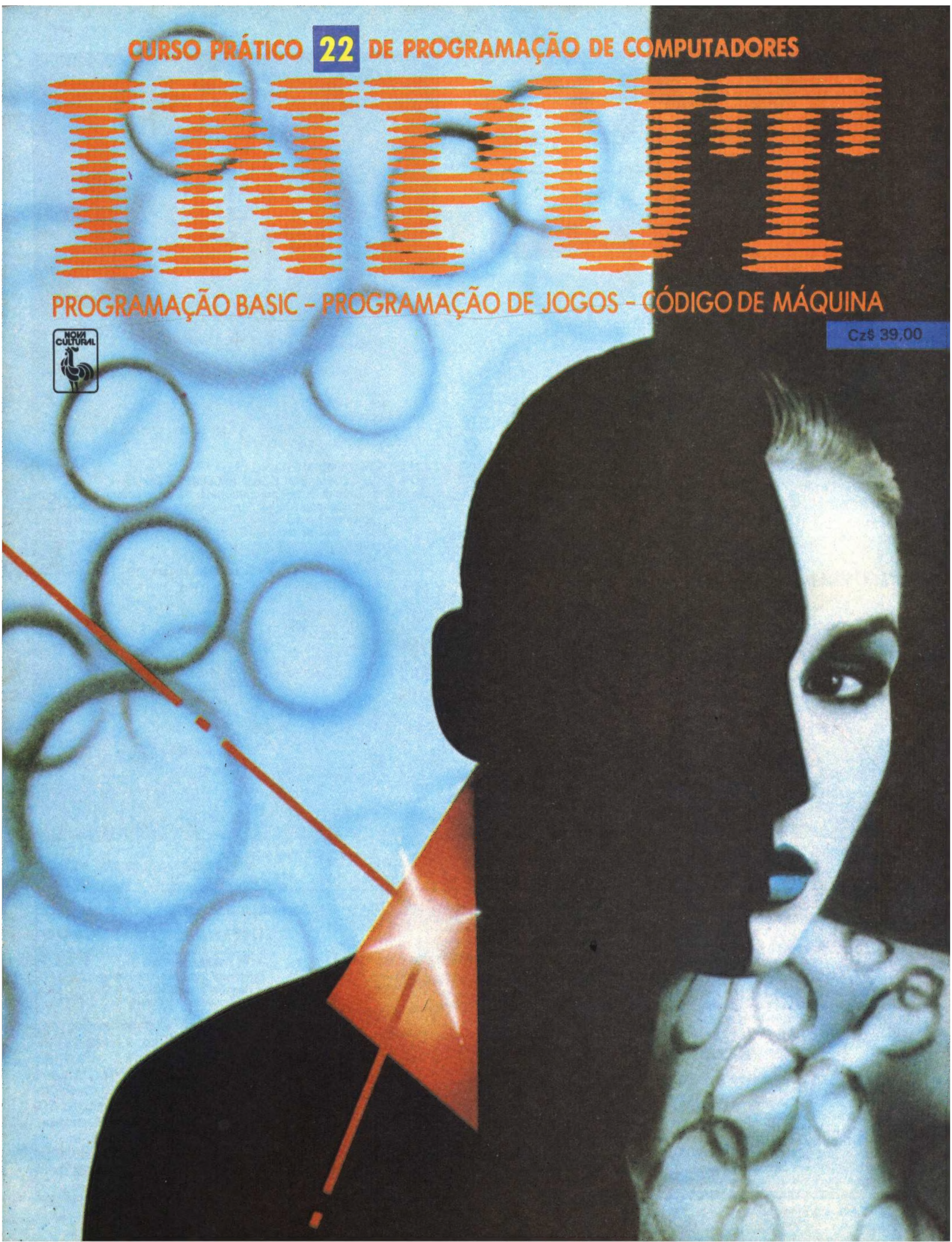
CURSO PRÁTICO 22 DE PROGRAMAÇÃO DE COMPUTADORES

INFO

PROGRAMAÇÃO BASIC - PROGRAMAÇÃO DE JOGOS - CÓDIGO DE MÁQUINA

NOVA CULTURAL

Cz$ 39,00



INPUT

Vol. 2

NESTE NÚMERO

APLICAÇÕES

ROTINAS PARA O CAD

Novas possibilidades do programa CAD (Desenho Assistido por Computador). Opções sofisticadas. Desenhos coloridos no Spectrum 421

PROGRAMAÇÃO BASIC

EDIÇÃO DE PROGRAMAS NO MSX

Edição de linhas. O controle do cursor. Teclas fundamentais e movimentos mais complexos. Como consolidar as modificações feitas 425

PROGRAMAÇÃO DE JOGOS

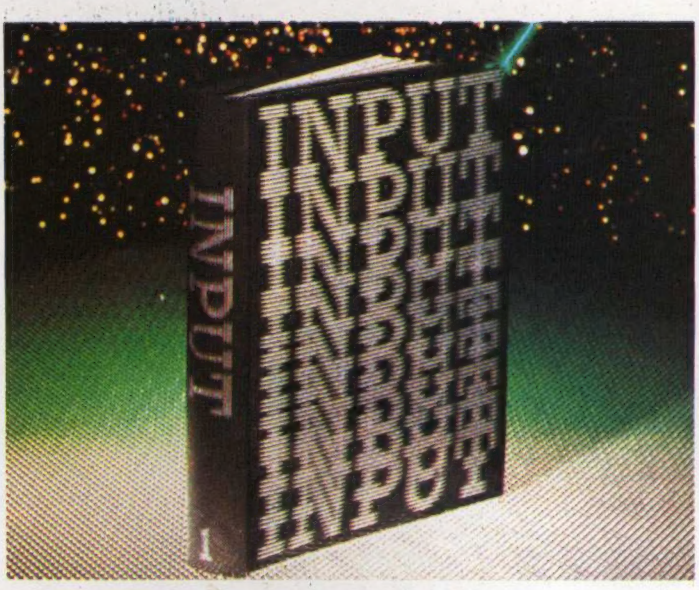
PROGRAME UM CARTEADO

Como desenhar as cartas do baralho. Posicionamento dos naipes. O uso do **VPOKE** na tela gráfica. Como embaralhar e distribuir as cartas... 426

PROGRAMAÇÃO BASIC

FUNÇÕES MATEMÁTICAS

A função de potenciação. A raiz quadrada. O uso da função **SQR** num programa..... 434



PLANO DA OBRA
INPUT é uma obra editada em fascículos semanais, e cada conjunto de 15 fascículos compõe um volume. A capa para encadernação de cada volume estará à venda oportunamente.

FÉRIAS, VIAGENS, MUDANÇAS...
NÃO FIQUE COM A COLEÇÃO INCOMPLETA
Se você está saindo de férias, pretende viajar ou vai se ausentar por algum tempo, avise antecipadamente seu jornaleiro. Ele pode guardar os seus fascículos enquanto você estiver fora. Se, por qualquer motivo, você perdeu alguns números, peça-os também a seu jornaleiro, ou entre em contato com nossa Distribuidora:

1. **Pessoalmente** — Em *São Paulo*, os endereços são: rua Brigadeiro Tobias, 773, Centro; av. Industrial, 117, Santo André. No *Rio de Janeiro*, av. Mem de Sá, 191/193, Centro.
2. **Por carta** — Envie para:
   DINAP — Distribuidora Nacional de Publicações
   Números Atrasados
   Estrada Velha de Osasco, 132 — Jardim Teresa
   CEP 06040 — Osasco — SP
3. **Por telex** — Utilize o nº (011) 33 670 DNAP.

Em **Portugal**, os pedidos devem ser feitos à Distribuidora Jardim de Publicações Lda. — Qta. Pau Varais, Azinhaga de Fetais, 2685, Camarate, Lisboa; Apartado 57; Telex 43 069 JARLIS P.
Atenção: Após seis meses do encerramento da coleção, o atendimento dos pedidos dependerá da disponibilidade do estoque.
Obs.: Quando pedir livros, mencione sempre o título e/ou autor da obra, além do número da edição.

COLABORE CONOSCO
Encaminhe seus comentários, críticas, sugestões ou reclamações ao
SERVIÇO DE ATENDIMENTO AO LEITOR
Caixa Postal 9 442, São Paulo — SP.

EDITOR
RICHARD CIVITA
NOVA CULTURAL

Presidente
Flávio Barros Pinto
Diretoria
Carmo Chagas, Iara Rodrigues,
Pierluigi Bracco, Plácido Nicoletto,
Roberto Silveira, Shozi Ikeda,
Sônia Carvalho

REDAÇÃO
Diretor Editorial: Carmo Chagas
Editores Executivos:
Antonio José Filho, Berta Sztark Amar
Editor Chefe: Paulo de Almeida
Editoras Assistentes: Ana Lúcia B. de Lucena,
Marisa Soares de Andrade
Chefe de Arte: Carlos Luiz Batista
Assistentes de Arte: Dagmar Bastos Sampaio,
Grace Alonso Arruda, Monica Lenardon Corradi

Secretária de Redação/Coordenadora: Stefania Crema
Secretário de Redação: Mauro de Queiroz

Colaboradores
Consultor Editorial Responsável:
Dr. Renato M. E. Sabbatini
(Diretor do Núcleo de Informática Biomédica da Universidade Estadual de Campinas-SP)
Execução Editorial: DATAQUEST Assessoria em Informática Ltda., Campinas
Tradução, adaptação, programação e redação:
Abílio Pedro Neto, Aluísio J. Dornellas de Barros, Marcelo R. Pires Therezo, Marcos Huascar Velasco, Raul Neder Porrelli, Ricardo J. P. de Aquino Pereira
Coordenação Geral: Rejane Felizatti Sabbatini

COMERCIAL
Diretor Comercial: Roberto Silveira
Gerente Comercial: Joaquim Celestino da Silva
Gerente de Circulação: Denise Mozol
Gerente de Propaganda e Publicidade: José Carlos Madio
Gerente de Pesquisa e Análise de Mercado:
Wagner M. P. Nabuco de Araújo

CLC

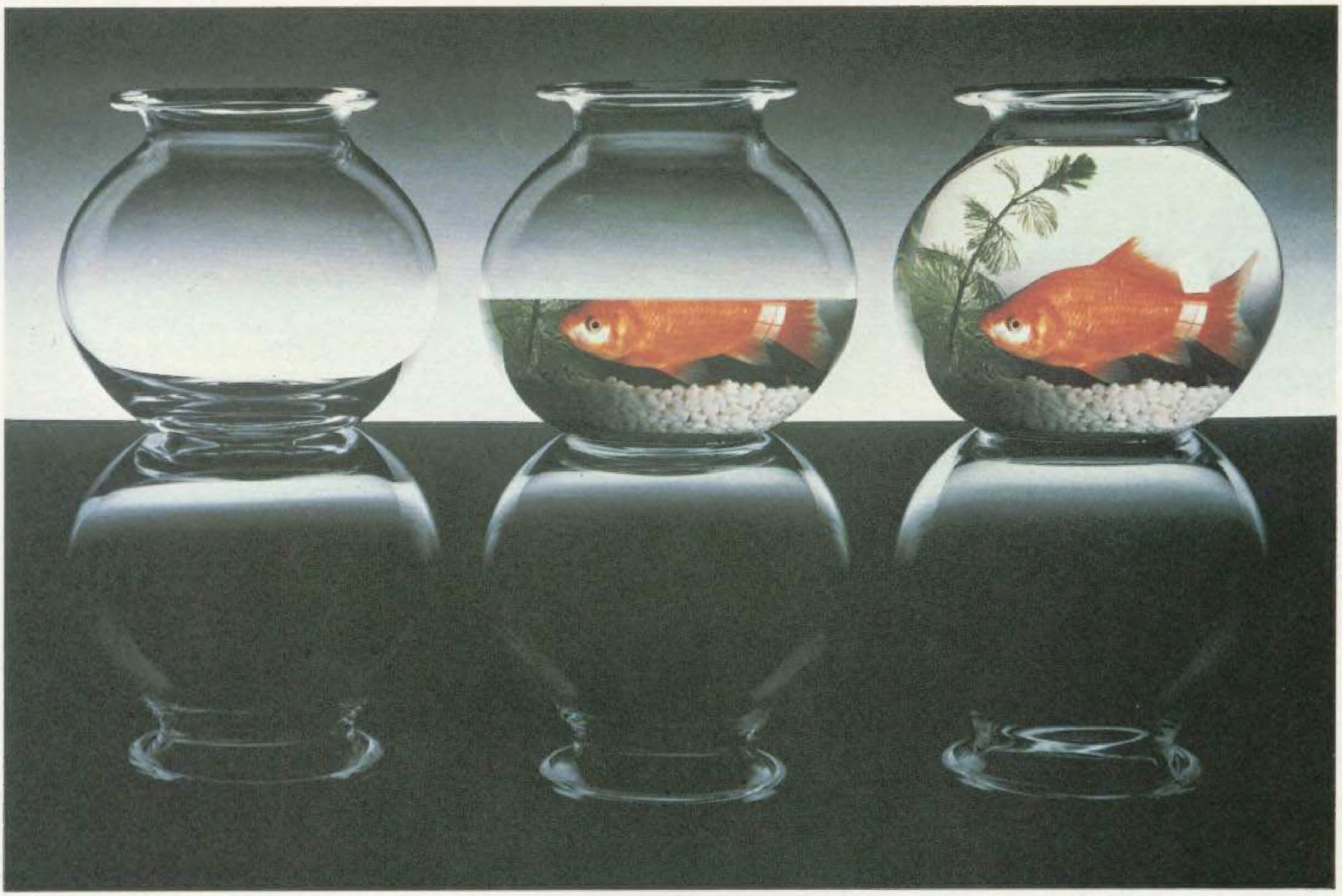
A Editora Nova Cultural Ltda. é uma empresa do Grupo CLC — Comunicações, Lazer e Cultura
Presidente: Richard Civita
Diretoria: Flávio Barros Pinto, João Gomez, Menahen M. Politi, Renê C. X. Santos, Stélio Alves Campos

© Marshall Cavendish Limited, 1984/85.
© Editora Nova Cultural Ltda., São Paulo, Brasil, 1986; 2ª edição, 1987.

Edição organizada pela Editora Nova Cultural Ltda.
Av. Brigadeiro Faria Lima, 2000 - 3º andar
CEP 01452 - São Paulo - SP - Brasil
(Artigo 15 da Lei 5 988, de 14/12/1973).
Esta obra foi composta pela AM Produções Gráficas Ltda. e impressa pela Companhia Lithographica Ypiranga.

ROTINAS PARA O CAD

- NOVAS POSSIBILIDADES DO PROGRAMA CAD
- OPÇÕES SOFISTICADAS
- CORRIGIR E COPIAR
- CORES NO SPECTRUM



Adicione rotinas especiais ao seu programa CAD (Desenho Assistido por Computador) e veja como tirar o melhor proveito da sofisticação que elas proporcionam à listagem original.

Com o programa CAD (Desenho Assistido por Computador), apresentado na página 414, vimos colocar os recursos gráficos do computador sob controle direto do teclado, o que nos permitiu fazer alguns desenhos bem sofisticados. Se quisermos, porém, explorar todo o potencial do programa — especialmente a possibilidade de colorir — deveremos examinar várias outras funções. O menu já nos deu uma idéia das diferentes opções, mas ainda não tivemos acesso a elas.

Carregue o programa anterior e adicione as linhas fornecidas neste artigo. Cada rotina é seguida de notas e sugestões de como tirar o melhor proveito das novas possibilidades do seu programa.

```
4000 REM retangulo e caixa
4010 LET caixa=0: GOTO 4030
4020 LET caixa=1
4030 FOR n=1 TO 50: NEXT n
4040 GOSUB 8000
4050 IF INKEY$=CHR$ 13 THEN  RA
ND USR 65380: GOTO 1000
4060 IF INKEY$<>CHR$ 32 THEN  G
OTO 4040
4070 FOR n=1 TO 50: NEXT n
4080 LET xx=0: LET yy=0: LET hx
=x: LET hy=y
4090 GOSUB 8000: FOR n=1 TO 2:
PLOT hx,hy
4100 DRAW  OVER 1;0,yy: DRAW  O
VER 1;xx,0: DRAW  OVER 1;0,-yy:
 DRAW  OVER 1;-xx,0: NEXT n
4110 IF INKEY$<>CHR$ 32 THEN  G
OTO 4090
4120 PLOT hx,hy: DRAW 0,yy: DRA
W xx,0: DRAW 0,-yy: DRAW -xx,0
4130 IF caixa=0 THEN  GOTO 4030
4135 IF xx=0 THEN  GOTO 4040
4140 FOR n=hx TO hx+xx STEP SGN
 xx
4150 PLOT n,hy: DRAW 0,yy: NEXT
 n
4160 GOTO 4040
5000 REM circulo
5010 FOR n=1 TO 50: NEXT n
5020 GOSUB 8000
5030 IF INKEY$=CHR$ 13 THEN  RA
```

```
ND USR 65380: GOTO 1000
5040 IF INKEY$<>CHR$ 32 THEN  G
OTO 5020
5050 FOR n=1 TO 50: NEXT n
5060 LET xx=0: LET yy=0: LET hx
=x: LET hy=y
5070 GOSUB 8000: CIRCLE  OVER 1
;hx,hy,ABS xx: CIRCLE  OVER 1;h
x,hy,ABS xx
5080 IF INKEY$<>CHR$ 32 THEN  G
OTO 5070
5090 CIRCLE hx,hy,ABS xx: GOTO
5000
5500 REM apagar
5510 GOSUB 8000
5520 IF POINT (x,y)=1 THEN  PLO
T  OVER 1;x,y
5530 IF INKEY$=CHR$ 13 THEN  RA
ND USR 65380: GOTO 1000
5540 GOTO 5510
6000 REM copiar
6010 COPY : GOTO 1000
7000 INPUT "INTRODUZA O NOME ";
 LINE n$: IF LEN n$>10 THEN  GO
TO 7000
7010 LOAD n$CODE 50000: RAND US
R 65368: GOTO 1000
7500 INPUT "INTRODUZA O NOME ";
 LINE n$: IF n$="" OR LEN n$>10
 THEN  GOTO 7500
7510 SAVE n$SCREEN$ : GOTO 1000
```

Selecione a opção RETANGULO e mova o cursor até onde deseja colocar um canto da figura que pretende desenhar; em seguida, pressione <**SPACE**>. Desloque o cursor para o canto diagonalmente oposto. Enquanto executa o movimento, um retângulo ficará piscando, para lhe dar a idéia exata do que está desenhando. Você pode, assim, escolher o tamanho e a forma ideal, simplesmente passeando com o cursor. Quando tudo estiver do seu agrado, pressione <**SPACE**>, para fixar o desenho.

A opção CAIXA funciona da mesma maneira que RETANGULO, só que a área interna é preenchida.

CIRCULO é a outra forma que você pode desenhar. Coloque o cursor no local que será o centro do círculo e pressione <**SPACE**>. Em seguida, mova-o para qualquer ponto da periferia e pressione novamente a tecla de espaço.

A opção COPIAR envia para sua impressora a imagem da tela. Depois de selecioná-la, responda às perguntas apresentadas. Terminado o trabalho, o programa retornará ao menu.

Para fazer correções e mudanças, use as opções APAGAR ou OOPS. Pequenos detalhes podem ser corrigidos com a primeira opção. Quando terminar, tecle <**ENTER**> para retornar ao menu.

Para mudanças radicais, utilize OOPS, que apagará tudo o que foi feito depois de sua última visita ao menu.

Os dois itens restantes são GRAVAR e CARREGAR (na fita, somente). Ao selecionar um dos dois, um nome será solicitado. Você pode CARREGAR sem especificar um nome de arquivo, simplesmente pressionando <**ENTER**>, mas, para GRAVAR, o nome é sempre necessário.

```
4000 SCREEN 1,ST:GOSUB 1500
4010 IF EF=1 GOSUB 500:RETURN
4020 IF PEEK(345)<>PC THEN 4000
4030 XS=X:YS=Y
4040 GOSUB 1500:GOSUB 500
4060 IF EF=1 THEN RETURN
4070 IF ABS((XS-X)*(YS-Y))>2300
0 THEN 4040
4080 LINE(X,Y)-(XS,YS),PSET,B
4090 IF PEEK(345)<>PC THEN 4040
4100 PMODE MD,5:GET(X,Y)-(XS,YS
),CP,G:PMODE MD,1
4110 XS=XS-X:YS=YS-Y
4120 GOSUB 1500:GOSUB 500
4130 IF EF=1 THEN RETURN
4140 IF(X+XS)<0 OR (X+XS)>255 O
R(Y+YS)<0 OR (Y+YS)>191 THEN 41
20
4150 LINE(X,Y)-(X+XS,Y+YS),PSET
,B
4160 IF PEEK(345)<>PC THEN 4120
4170 GOSUB 500:PUT(X,Y)-(X+XS,Y
+YS),CP,PSET:GOSUB 510
4180 GOTO 4120
5000 CLS:PRINT" SELECIONE A COR
 DA BORDA (0-8) "
5010 A$=INKEY$:IF A$<"0" OR A$>
"8" THEN 5010
5020 BC=VAL(A$):SCREEN 1,ST
5030 GOSUB 1500:GOSUB 500
5040 IF EF=1 THEN RETURN
5050 IF PEEK(345)<>PC THEN 5030
5060 PAINT(X,Y),CL,BC:GOSUB 510
:GOTO 5030
6000 CLS:PRINT" SALVAR OU CARRE
GAR DO GRAVADOR (S/C) ?"
6010 A$=INKEY$:IF A$<>"S" AND A
$<>"C" THEN 6010
6020 IF A$="S" THEN 6100
6030 PRINT" CONFIRMA QUE QUER C
ARREGAR      OUTRO DESENHO (S/N
)?"
6040 A$=INKEY$:IF A$<>"N" AND A
$<>"S" THEN 6040
6050 IF A$="N" THEN RETURN
6060 MOTORON:PRINT:PRINT " POSI
CIONE O TAPE,APERTE 'PLAY'  E E
NTAO PRESSIONE <ENTER>"
6070 IF INKEY$<>CHR$(13) THEN 6
070
6080 MOTOROFF:PRINT:PRINT:INPUT
"INTRODUZA O NOME DO ARQUIVO ";
A$
6090 SCREEN 1,ST:CLOADM A$:GOSU
B 510:RETURN
6100 PRINT:INPUT"NOME DO ARQUIV
O ";A$
6110 MOTORON:PRINT:PRINT" POSIC
IONE O TAPE,APERTE'RECORD' E EN
TAO PRESSIONE <ENTER>"
6120 IF INKEY$<>CHR$(13) THEN 6
120
6130 GOSUB 500:CSAVEM A$,1536,7
679,1536:RETURN
```

A opção RETANGULO permite que você desenhe retângulos de várias cores, em qualquer lugar da tela. Selecione a opção e mova o cursor até o ponto da tela onde você quer um canto da figura. Pressione a barra de espaço para identificar o ponto. Mova o cursor para o canto oposto. Da mesma maneira que com DESENHO e LINHA, você pode escolher cores ou abandonar a opção pressionando <**ENTER**>. Quando estiver satisfeito com o desenho, pressione novamente a barra de espaço.

CAIXA funciona como a opção anterior, mas a figura é preenchida com a cor em uso.

CIRCULO e DISCO são como RETANGULO e CAIXA. O primeiro ponto escolhido fica na periferia da forma. Mova o cursor, até chegar ao ponto desejado, que é o centro.

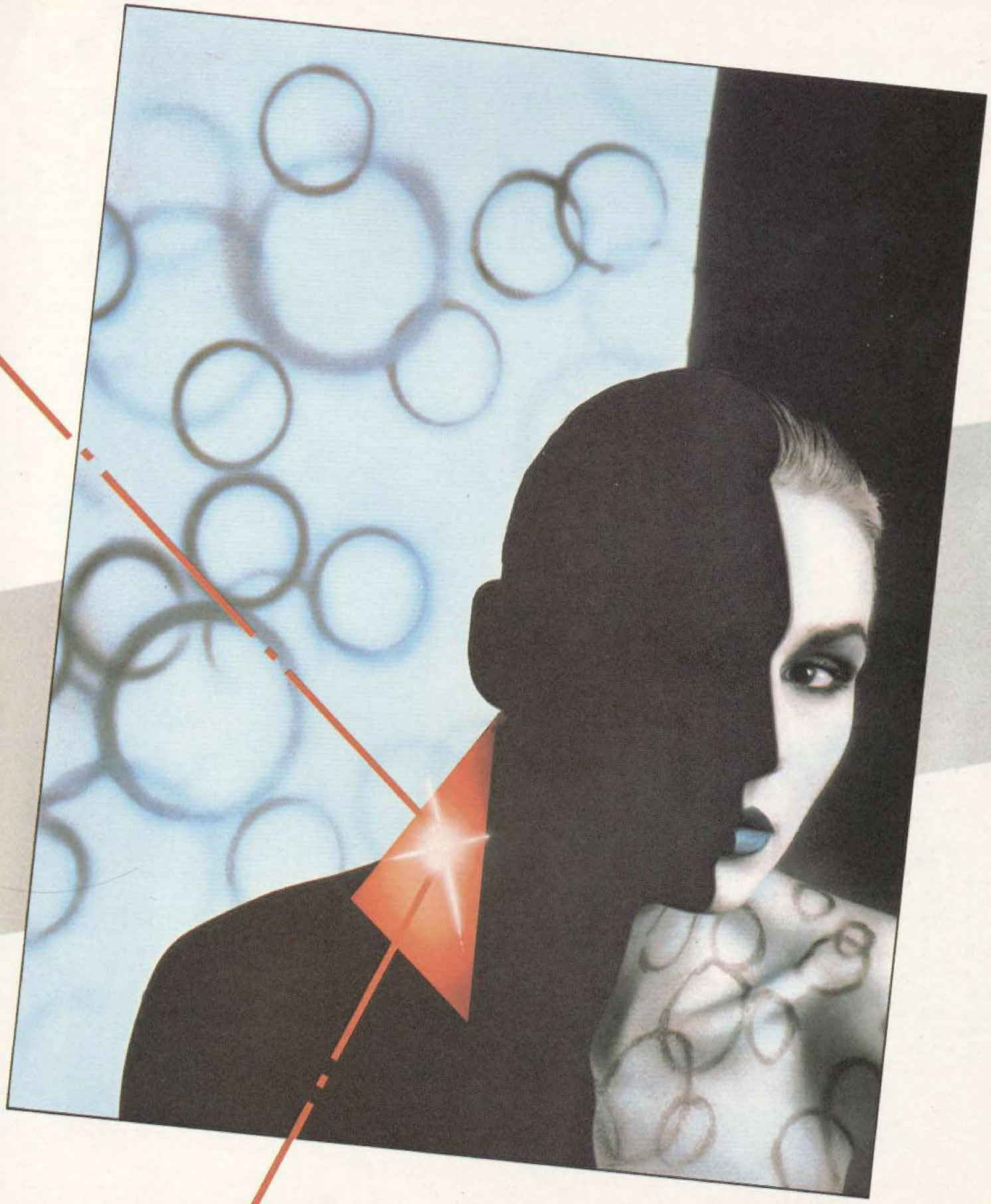
ELIPSE difere um pouco de CIRCULO. O primeiro ponto escolhido é o centro. Ao mover o cursor em qualquer direção, verá apenas uma linha. Mas, ao movê-lo na horizontal e depois na vertical, por exemplo, uma elipse começará a crescer. Ajuste-a até obter o que deseja e pressione a barra de espaço.

A opção COPIAR permite que você duplique uma imagem já desenhada na tela em outra parte dela. Para usá-la, posicione o cursor em um canto da área a ser copiada e pressione a barra de espaço. Ao mover o cursor em direção ao canto oposto, verá um retângulo que demarca a região onde o desenho será duplicado. Determine a área, quando estiver satisfeito com o seu tamanho e forma, pressionando novamente a barra de espaço. Mova essa área até o ponto desejado, usando as setas. Pressione a barra de espaço para efetuar a cópia, apagando o que havia naquela área. Até metade da tela pode ser copiada, sem problemas.

A opção PREENCHER funciona como o comando **PAINT** em BASIC. Ao selecioná-la, você deve informar a cor dos limites onde o **PAINT** deve parar. Mova, então, o cursor até a área a ser preenchida e pressione a barra de espaço. Escolha a cor pelas teclas 0 a 8.

Para corrigir possíveis erros, selecione a opção ERRO, que apaga tudo o que foi feito desde sua última visita ao menu. Pode-se, também, corrigir detalhes, selecionando a cor de fundo e apagando linhas ou pontos. Áreas maiores devem ser corrigidas com CAIXA ou DISCO.

A última opção permite que desenhos sejam carregados ou gravados na fita: conecte o cassete e responda às perguntas apresentadas.



```
3070  HPLOT PX(1),PY(1) TO PX(2
),PY(1) TO PX(2),PY(2) TO PX(1)
,PY(2) TO PX(1),PY(1):M = 0: GO
TO 3000
3080 RA =  SQR ((PX(1) - PX(2))
 * (PX(1) - PX(2)) + (PY(1) - P
Y(2)) * (PY(1) - PY(2))): FOR I
 = 0 TO 2 * PI STEP PI / 180
3090  HPLOT PX(2) + RA *  SIN (
I),PY(2) - RA *  COS (I)
3100  NEXT
3110  GOTO 3000
4000 CI = 0
4010  GOSUB 1500: GOSUB 1000: I
F A$ <  > CHR$ (32) THEN  GOTO
 4010
4020  POKE M,ME
4030 CI = CI + 1:PX(CI) = X:PY(
CI) = Y: PRINT  CHR$ (7);: HPLO
T X,Y:M = 0
4040  IF CI <  > 3 THEN 4010
4050 CE =  SQR ((PX(1) - PX(2))
 * (PX(1) - PX(2)) + (PY(1) - P
Y(2)) * (PY(1) - PY(2))):CE = C
E / 2
4060 A1 =  SQR ((PX(1) - PX(3))
 * (PX(1) - PX(3)) + (PY(1) - P
Y(3)) * (PY(1) - PY(3)))
4070 A2 =  SQR ((PX(3) - PX(2))
 * (PX(3) - PX(2)) + (PY(3) - P
Y(2)) * (PY(3) - PY(2))):AE = (
A1 + A2) / 2
4080 BE =  SQR (AE * AE - CE *
CE)
4090  FOR I = 0 TO 2 * PI STEP
PI / 180
4100  IF PX(1) <  > PX(2) THEN
4120
4110  HPLOT PX(1) + BE *  COS (
I),PY(1) + ( - 1 ^ (PY(1) > PY(
2))) * CE - AE *  SIN (I): NEXT
 : GOTO 4000
4120  HPLOT PX(1) + ( - 1 ^ (PX
(1) > PX(2))) * CE + AE *  COS
(I),PY(1) - BE *  SIN (I): NEXT
 : GOTO 4000
7000  TEXT : HOME : PRINT "Grav
ar/Carregar"
7010  VTAB 10
7020  INPUT "Opcao? (G/C) ";R$
7030  IF R$ <  > "G" AND R$ <
> "C" THEN 7030
7040  INPUT "Nome do desenho? "
;DE$
7050  INPUT "Pagina? ";P
7060  IF R$ = "C" THEN 7080
7070  PRINT  CHR$ (4);"BSAVE";D
E$;",A";P * 8192;",L";8192: GOT
O 250
7080  PRINT  CHR$ (4);"BLOAD";D
E$;",A";P * 8192: GOTO 250
```

Usaremos agora as opções mais sofisticadas do nosso editor de desenho. RETANGULO, CIRCULO e ELIPSE estão à sua disposição.

Para a opção RETANGULO, mova o cursor até um canto do quadrilátero que deseja traçar. Tecle a barra de espaço, leve o cursor para o canto diagonalmente oposto e tecle novamente a barra de espaço. Sua figura está pronta.

CIRCULO funciona de maneira parecida, só que o primeiro ponto corresponde ao centro da figura a ser traçada. O segundo determina o tamanho dela, ficando na periferia.

Ao usar a opção ELIPSE, não se esqueça de que o maior eixo da figura deve estar sempre na horizontal ou na vertical. Selecione três pontos para traçá-la. Os dois primeiros funcionarão como os focos da elipse e o terceiro ficará na periferia.

Por fim, GRAVAR/CARREGAR permite que você grave uma imagem em disquete ou que a carregue na memória do micro. É possível, inclusive, carregar um desenho pronto de algum outro programa ou jogo. A identificação de uma imagem de alta resolução no diretório é fácil: em geral, ela ocupa 34 setores do disco e tem a letra B à frente do nome do arquivo. Para carregá-la, simplesmente coloque o disquete no drive e digite o nome correto do arquivo quando este for solicitado pelo programa.

```
480 CI=0
490 GOSUB310:GOSUB210:IFA$<>CHR
$(32)THEN490
500 CI=CI+1:PX(CI)=X:PY(CI)=Y:B
EEP:PSET(X,Y),CO:M=0
510 IFCI<>2THEN490
520 ONOPGOTO590,590,530,540,550
,560,560
530 LINE(PX(1),PY(1))-(PX(2),PY
(2)),CO:M=0:GOTO480
540 LINE(PX(1),PY(1))-(PX(2),PY
(2)),CO,B:M=0:GOTO480
550 LINE(PX(1),PY(1))-(PX(2),PY
(2)),CO,BF:M=0:GOTO480
560 RA=SQR((PX(1)-PX(2))*(PX(1)
-PX(2))+(PY(1)-PY(2))*(PY(1)-PY
(2)))
570 CIRCLE (PX(1),PY(1)),RA,CO:
M=0:IFOP=6THENGOTO480
580 PAINT (PX(1)+1,PY(1)),CO:GO
TO480
590 LINE(PX(1),PY(1))-(PX(2),PY
(2)),CL,BF:M=0:GOTO480
600 CI=0
610 GOSUB310:GOSUB210:IFA$<>CHR
$(32)THEN610
620 CI=CI+1:PX(CI)=X:PY(CI)=Y:B
EEP:PSET(X,Y),CO:M=0
630 IFCI<>3THEN610
640 CE=SQR((PX(1)-PX(2))*(PX(1)
-PX(2))+(PY(1)-PY(2))*(PY(1)-PY
(2))):CE=CE/2
650 C1=SQR((PX(1)-PX(3))*(PX(1)
-PX(3))+(PY(1)-PY(3))*(PY(1)-PY
(3)))
660 C2=SQR((PX(3)-PX(2))*(PX(3)
-PX(2))+(PY(3)-PY(2))*(PY(3)-PY
(2))):AE=(C1+C2)/2
670 BE=SQR(AE*AE-CE*CE)
680 FORJ=0TO2*PI STEP PI/90
690 IFPX(1)<>PX(2)THEN720
700 IFPY(1)<PY(2)THENS=1ELSES=-
1
710 PSET (PX(1)+BE*COS(J),PY(1)
+S*CE-AE*SIN(J)),CO:NEXT:GOTO60
0
720 IFPX(1)<PX(2)THENS=1ELSES=-
1
730 PSET (PX(1)+S*CE+AE*COS(J),
PY(1)-BE*SIN(J)),CO:NEXT:GOTO60
0
```

Apagar permite que você suprima o desenho de determinadas áreas da tela. Selecione a opção e coloque o cursor no canto do retângulo que corresponde à área que pretende apagar. Pressione a barra de espaço. Leve o cursor para o canto diagonal oposto e pressione novamente a barra de espaço. Tudo o que estiver dentro do retângulo formado pelos 2 pontos será apagado. Tecle <**RETURN**> para trocar sua opção.

Retângulo e **Caix**a funcionam de maneira semelhante, sendo que a primeira opção desenha uma figura vazia, ao contrário da segunda. Ao selecionar uma delas, leve o cursor a um extremo da figura e tecle a barra de espaço. Desloque-o até a outra extremidade e repita a operação. Sua figura será desenhada instantaneamente.

Circulo e **Disc**o têm uma relação semelhante à das duas opções anteriores. O primeiro ponto será o centro da figura e o segundo determinará o tamanho dela, ficando na periferia.

Elipse requer três pontos para cumprir sua função. Os dois primeiros determinam os focos da elipse e o terceiro, um ponto da periferia. Essa figura não é preenchida por cor. Para desenhos coloridos, você deverá recorrer à opção Pintar.

MICRO DICAS

CORES NO SPECTRUM

O programa apresentado permite que você faça desenhos coloridos no Spectrum, mas é preciso ter cuidado para evitar problemas de sobreposição de cores. Se duas ou mais cores forem colocadas no mesmo retângulo de caractere, a última tomará o lugar das outras. Planeje seu desenho de maneira que cores adjacentes sejam alinhadas de acordo com esses retângulos. Lembre-se de que em cada linha temos 32 retângulos que contêm oito subdivisões na horizontal e 22 retângulos com oito subdivisões na vertical.

EDIÇÃO DE PROGRAMAS NO MSX

- COMO EDITAR LINHAS DE UM PROGRAMA BASIC
- O CONTROLE DO CURSOR
- TECLAS FUNDAMENTAIS E MOVIMENTOS MAIS COMPLEXOS

Os micros da linha MSX possuem sofisticados recursos de edição. Suas teclas de controle e seu cursor tornam possível usar o vídeo como se fosse uma verdadeira "folha de papel".

Todos os interpretadores da linguagem BASIC oferecem alguns recursos para a edição de programas, ou seja, técnicas de entrada e alteração das linhas que os compõem. O primeiro interpretador BASIC, desenvolvido na década de 60 na Universidade de Dartmouth, nos EUA, fixou os recursos mais elementares de edição: numeração, inserção, apagamento e listagem de linhas. Estes foram posteriormente adotados em todos os outros dialetos do BASIC que surgiram.

O passo seguinte consistiu em dar ao programador a possibilidade de alterar caracteres, já que, com os recursos anteriores, o erro de um caractere obrigava-o a digitar novamente a linha toda.

Com esse objetivo, desenvolveram-se comandos como o **EDIT**, presentes nos micros da linha Sinclair, TRS-80 e TRS-Color. Embora os recursos de edição disponíveis no **EDIT** sejam bastante versáteis, eles ainda apresentam uma limitação: operam apenas em uma linha — cujo número precisa ser explicitado — de cada vez.

EDIÇÃO BIDIMENSIONAL

A limitação mencionada não passa de um resquício do tempo em que o BASIC era operado através de terminais impressores, desprovidos de movimentação bidimensional do cursor.

Os projetistas do MSX, porém, souberam superar essa limitação, abrindo a possibilidade de se editar um programa inteiro, e não uma linha de cada vez, desde que o trecho a ser editado estivesse exibido na tela. O comando **EDIT** foi, assim, suprimido.

O conceito de edição de programas no MSX aproxima-se, incidentalmente, do utilizado nos avançadíssimos micros profissionais da linha PC: envolve a utilização de teclas especiais, ou da combinação de determinadas teclas, para "passear" o cursor de texto pela tela, inserir e apagar caracteres em qualquer ponto da mesma, etc. O computador "lembra-se" das modificações realizadas na página de vídeo desde que se pressione a tecla <**ENTER**> ou <**RETURN**> após o término das modificações em uma determinada linha.

Uma possibilidade interessante do MSX é a de editar também os números de linha. Quando se utiliza esse recurso, obtém-se duas linhas: a que tinha o número original, em sua própria posição, e outra — que pode ser uma duplicata da anterior ou incluir modificações —, que é automaticamente inserida em sua nova posição.

As teclas fundamentais para a edição no MSX são as seguintes:

→ - desloca o cursor de texto em uma posição para a direita

← - desloca o cursor de uma posição para a esquerda

↑ - desloca o cursor para cima

◁ - apaga o caractere imediatamente anterior ao cursor e recua o cursor de uma posição (retrocesso)

↓ - desloca o cursor para baixo

<**INS**> - ativa ou desativa o modo de inserção de caracteres

<**DEL**> - apaga um caractere e rejunta o restante da linha

<**ENTER**> - consolida as modificações realizadas em uma linha

A tecla **HOME** também pode ser utilizada durante o processo de edição, para fazer o cursor retornar à posição no topo esquerdo da tela.

Ao editar um programa, use primeiro o comando **LIST**, para colocar na tela o trecho do programa que deseja editar. Em seguida, desloque o cursor até a linha a ser editada. Para fazer as modificações, escreva por cima do texto exibido, ou insira e apague caracteres por meio das teclas **INS** e **DEL**. Em seguida, pressione a tecla <**ENTER**> ou <**RETURN**>, se quiser preservar as modificações feitas.

A tecla **INS** em geral está desativada — ou seja, se você pressionar qualquer tecla de caractere, ele será impresso sobre o que estiver sob o cursor no momento. Para inserir um ou mais caracteres em determinado ponto de uma linha, primeiro desloque o cursor até lá. Em seguida, pressione uma vez a tecla **INS**, colocando o computador em modo de inserção. Para encerrar o processo, pressione **INS** novamente ou, então, <**ENTER**>.

Para verificar se o computador está em modo de inserção, observe o cursor de texto: de um retângulo, ele se transforma em um traço pequeno. Se você pressionar a tecla **DEL** durante o modo de inserção, poderá apagar caracteres sem desligar a inserção.

Com o auxílio da tecla <**CONTROL**>, pode-se obter movimentos mais complexos do cursor, assim como o acionamento de funções adicionais de edição. Pressionando-se simultaneamente <**CONTROL**> e uma outra tecla alfabética, chega-se a alguns resultados interessantes:

<**CONTROL**><**F**> - desloca o cursor para a primeira letra da próxima palavra da linha

<**CONTROL**><**B**> - desloca o cursor para a primeira letra da palavra anterior, na linha

<**CONTROL**><**N**> - desloca o cursor para o final da linha

<**CONTROL**><**E**> - apaga a linha corrente desde o ponto onde está o cursor até o final

<**CONTROL**><**U**> - apaga toda a linha onde está o cursor

Convém lembrar, finalmente, que os recursos "normais" de edição de programas (numeração, inserção de linhas, renumeração, listagem, apagamento, etc.), presentes no editor padrão do BASIC, também funcionam no MSX.

PROGRAME UM CARTEADO

Microcomputadores podem se revelar grandes jogadores de baralho, com a vantagem de que nunca se cansam de jogar. Veja aqui como produzir gráficos para um carteado.

Seus amigos e parentes se negam a jogar baralho com você? Já se cansaram de perder dinheiro? O cacife é muito alto para seu orçamento? A solução para qualquer um desses problemas pode estar na série de artigos que iniciamos aqui. Programando seu micro para jogar Vinte-e-um, você terá uma vítima perfeita e poderá jogar sem perder um níquel.

Nesta primeira seção, veremos como programar os gráficos necessários para criar um baralho na memória — e na tela — de seu computador. O restante do programa — o jogo propriamente dito — será apresentado nos dois próximos artigos de *Programação de Jogos*. Grave o programa por partes, à medida que for sendo ampliado.

Se você não sabe jogar Vinte-e-um, não se preocupe: explicaremos as regras na última seção da série. Antes, porém, você precisará de um baralho completo.

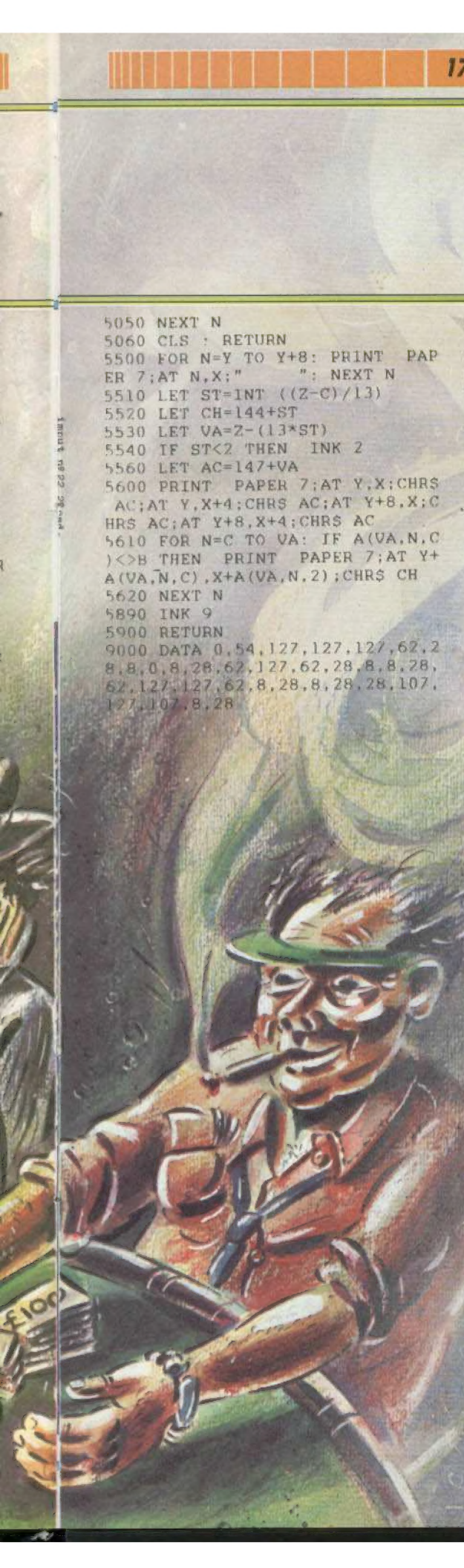
S

A rotina gráfica que possibilita ao computador mostrar as cartas na tela é a seguinte:

```
10 BORDER 4: PAPER 4: INK 9:
CLS : POKE 23658,8: LET B=0:
LET C=1
20 FOR N=USR "A" TO USR "R"+7
: READ A: POKE N,A: NEXT N
30 DIM C(52): FOR N=C TO 52:
LET C(N)=N: NEXT N
40 DIM A(13,13,2)
50 FOR N=C TO 10: FOR M=C TO
N: READ A(N,M,C),A(N,M,2):
NEXT M: NEXT N
60 FOR N=11 TO 13: LET A(N,C,
C)=4: LET A(N,C,2)=2: NEXT N
70 LET CC=C: LET CP=100
80 GOSUB 5000
500 LET Y=0: LET X=1
525 LET Z=C(CC)
530 GOSUB 5500
540 STOP
5000 CLS : PRINT AT 10,5;"EMBAR
ALHANDO AS CARTAS"
5010 FOR N=C TO 100
5020 LET X=INT (RND*52)+C
5030 LET Y=INT (RND*52)+C
5040 LET Z=C(X): LET C(X)=C(Y):
LET C(Y)=Z
```

- GRÁFICOS DE ALTA RESOLUÇÃO
- COMO DESENHAR AS CARTAS DO BARALHO
- POSICIONAMENTO DOS NAIPES
- COMO USAR **VPOKE** NA TELA GRÁFICA DO MSX
- COMO EMBARALHAR E DISTRIBUIR AS CARTAS

```
5050 NEXT N
5060 CLS : RETURN
5500 FOR N=Y TO Y+8: PRINT  PAP
ER 7;AT N,X;"      ": NEXT N
5510 LET ST=INT ((Z-C)/13)
5520 LET CH=144+ST
5530 LET VA=Z-(13*ST)
5540 IF ST<2 THEN  INK 2
5560 LET AC=147+VA
5600 PRINT  PAPER 7;AT Y,X;CHR$
 AC;AT Y,X+4;CHR$ AC;AT Y+8,X;C
HR$ AC;AT Y+8,X+4;CHR$ AC
5610 FOR N=C TO VA: IF A(VA,N,C
)<>B THEN  PRINT  PAPER 7;AT Y+
A(VA,N,C),X+A(VA,N,2);CHR$ CH
5620 NEXT N
5890 INK 9
5900 RETURN
9000 DATA 0,54,127,127,127,62,2
8,8,0,8,28,62,127,62,28,8,8,28,
62,127,127,62,8,28,8,28,28,107,
127,107,8,28
```



```
9010 DATA 0,8,20,34,34,62,34,34
,0,28,34,2,4,24,32,62
9020 DATA 0,28,34,2,12,2,34,28,
0,4,12,20,36,62,4,14
9030 DATA 0,62,32,32,60,2,34,28
,0,28,34,32,60,34,34,28
9040 DATA 0,62,34,2,4,8,16,16,0
,28,34,34,28,34,34,28
9050 DATA 0,28,34,34,30,2,34,28
,0,76,82,82,82,82,82,76
9060 DATA 0,14,4,4,4,4,4,36,24,0,
28,34,34,34,58,102,30,0,118,36,
40,48,40,36,118
9070 DATA 85,85,85,85,85,85,85,
85
9100 DATA 4,2
9110 DATA 2,2,6,2
9120 DATA 2,2,4,2,6,2
9130 DATA 1,1,1,3,7,1,7,3
9140 DATA 1,1,1,3,4,2,7,1,7,3
9150 DATA 1,1,1,3,4,1,4,3,7,1,7
,3
9160 DATA 1,1,1,3,2,2,4,1,4,3,7
,1,7,3
9170 DATA 1,1,1,3,2,2,4,1,4,3,6
,2,7,1,7,3
9180 DATA 1,1,1,3,3,1,3,3,4,2,5
,1,5,3,7,1,7,3
9190 DATA 1,1,1,3,2,2,3,1,3,3,5
,1,5,3,6,2,7,1,7,3
```

A linha 10 seleciona as cores da borda, dos caracteres e do fundo; **POKE** faz com que todas as letras sejam maiúsculas. As duas variáveis — **B** e **C** — são usadas no lugar de "0" e "1" no restante do programa. Devido à maneira como o Spectrum trabalha com números e variáveis, isso permitirá uma economia de 6 bytes, sempre que esses valores aparecerem. Assim, o programa poderá rodar no Spectrum de 16K.

A linha 20 prepara os caracteres (UDGs) usados nos naipes, números e letras de cada carta. Os dados para isso estão nas linhas **DATA** 9000 e 9070. Em seguida, a linha 30 cria um conjunto de 52 cartas não embaralhadas. A matriz **A**, dimensionada na linha 40, é preenchida com os valores das linhas **DATA** 9100 e 9190, que contêm as coordenadas das figuras dos naipes em cada carta. A linha 50 preenche parte da matriz com as coordenadas dos símbolos das cartas numéricas, enquanto a linha 60 preenche o restante com as coordenadas dos mesmos nas cartas com figuras. É possível criar desenhos para as cartas com figuras, mas seria bem cansativo e demorado digitar os dados necessários. Além disso, o programa ficaria grande demais para o Spectrum de 16K.

A linha 70 coloca os valores 1 e 100 nas variáveis **CC** e **CP**, respectivamente. **CC** é a carta atual, ou o elemento da matriz de cartas que o computador selecionou por último. **CP** é a quantidade de fichas que o jogador possui.

A linha 80 chama a sub-rotina que embaralha as cartas. Ela começa na linha 5000, que apaga a tela e comunica que as cartas estão sendo embaralhadas. O computador embaralha as cartas escolhendo duas, ao acaso, e trocando suas posições. O laço **FOR...NEXT** entre as linhas 5010 e 5050 repete o processo 100 vezes. Existe uma chance de que uma mesma carta seja escolhida "trocando de posição consigo mesma". Isso, na prática, não tem nenhuma importância, pois o número de trocas de posição é suficiente para garantir uma boa "embaralhada".

O valor da linha 5010 pode ser alterado para aumentar o número de trocas, mas números um pouco maiores que 100 já provocam uma demora inaceitável. A sub-rotina termina na linha 5060, que limpa a tela e retorna.

A linha 525 coloca na variável **Z** o valor da carta atual — elemento **CC** da matriz **C**. Em seguida, a linha 530 chama a sub-rotina 5500, que coloca as cartas na tela. A linha 5500 apresenta a parte branca da carta. A linha 5510 selecio-

na o naipe correto (os naipes são numerados de 0 a 3). A linha 5520 calcula o código do caractere que contém o símbolo do naipe escolhido. A linha 5530 identifica a carta — afinal, o número de vezes que o símbolo do naipe vai aparecer depende disso.

Antes de desenhar o naipe, o computador seleciona a cor adequada. A linha 5540 atribui a cor vermelha aos naipes com número 0 ou 1, e a preta aos outros dois.

A linha 5600 desenha o número da carta usando o caractere adequado, escolhido pela linha 5560. A linha 5610 desenha os símbolos do naipe, cujas coordenadas são obtidas na matriz A.

Finalmente, com a escolha da cor 9 (cor de contraste), termina a sub-rotina.

MSX

Sprites são mais adequados para mover figuras. Como as cartas ficarão paradas, convém utilizar outra técnica para desenhar os números, letras e naipes das cartas.

```
10 CLEAR 500:COLOR 15,12,12:SCR
EEN0:KEY OFF
20 LOCATE 8,11:PRINT "EMBARALHA
NDO AS CARTAS"
30 DIM NU$(13),NA(32):FOR K=1 T
O 13:READ NU$(K):NEXT
40 DATA BD2S4U5ER3FD2NL4D3,RDLD
R,RDNLDL,DRDU2,NRDRDL,D2RULUR,R
S8DGD,ND2RDNLDL,NDRDNLD,D2S10BR
S12NU2RU2L,S4BD5F2R2U6L3R4,S4NR
5D6R3NH2NFR2U6,DNDS8RS12NEF
50 FOR I=1 TO 32
60 NA(I)=VPEEK(BASE(2)+23+I)
70 NEXT I
82 DIM SQ(61)
84 R=RND(-TIME):MN=100
88 SCREEN 2:FOR I=0 TO 3
90 VPOKE BASE(10)+252+I,32
92 VPOKE BASE(10)+252+I+256,32
94 VPOKE BASE(10)+252+I+512,32
96 NEXT I
120 FOR I=1 TO 32
130 VPOKE BASE(12)+2015+I,NA(I)
131 VPOKE BASE(11)+2015+I,(ABS(
I<17)*5+1)*16+15
132 VPOKE BASE(12)+4063+I,NA(I)
133 VPOKE BASE(11)+4063+I,(ABS(
I<17)*5+1)*16+15
134 VPOKE BASE(12)+6111+I,NA(I)
135 VPOKE BASE(11)+6111+I,(ABS(
I<17)*5+1)*16+15
140 NEXT
160 FOR K=0 TO 51:SQ(K)=K:NEXT
180 GOSUB 1500:N=0
190 FOR CX=31 TO 200 STEP 48:FO
R CY=1 TO 140 STEP 104
200 GOSUB 1000:GOSUB 2000:FOR J
=1 TO 500:NEXT J,CY,CX
210 FOR J=1 TO 1000:NEXT:GOTO 1
90
1000 ST=INT(SQ(N)/13)+1:NM=SQ(N
)-13*ST+14:N=N+1:IF N>51 THEN N
=0
1010 RETURN
1500 FOR X=52 TO 2 STEP -1
1510 Q=INT(RND(1)*X)+1
1520 T=SQ(X-1):SQ(X-1)=SQ(Q-1):
SQ(Q-1)=T
1530 NEXT
1540 FOR X=0 TO 9:SQ(X+52)=SQ(X
):NEXT
1550 RETURN
2000 LINE (CX-1,CY-1)-(CX+42,CY
+71),15,BF
2005 LINE (CX-2,CY-1)-(CX-2,CY+
71),12
2010 S$=NU$(NM):PX=CX+24:PY=CY+
24:GOSUB 2550:PY=CY+15:GOSUB 25
50:PY=CY+35:GOSUB 2550:FOR J=0
```

```
TO 4:PX=CX+7:PY=CY+8+J*8:GOSUB
2550:PX=CX+34:GOSUB 2550:NEXT
2020 IF ST>2 THEN COLOR1 ELSE C
OLOR6
2030 DRAW "S12BM"+STR$(CX+3)+",
"+STR$(CY+2)+S$
2040 DRAW "S12BM"+STR$(CX+35)+"
,"+STR$(CY+62)+S$
2050 IF NM/2<>INT(NM/2) OR NM>1
0 THEN PX=CX+24:PY=CY+24:GOSUB
2500
2060 IF NM=2 OR NM=3 OR NM=10 O
R NM=8 THEN PX=CX+24:PY=CY+15:G
OSUB 2500:PY=PY+19:GOSUB 2500
2080 IF NM<4 OR NM>10 THEN 2140
2090 IF (NM=10 OR NM=8) THEN NS
=INT((NM-1)/2) ELSE NS=INT(NM/2
)
2100 FOR J=0 TO NS-1
2110 PX=CX+7:PY=CY+8+J*38/(NS-1
):GOSUB 2500
2120 PX=CX+34:GOSUB 2500
2130 NEXT
2140 RETURN
```

```
2500 ON ST GOTO 2510,2520,2530,
2540
2510 VPOKE BASE(10)+INT(PY/8)*3
2+INT(PX/8)+32,252:RETURN
2520 VPOKE BASE(10)+INT(PY/8)*3
2+INT(PX/8)+32,253:RETURN
2530 VPOKE BASE(10)+INT(PY/8)*3
2+INT(PX/8)+32,254:RETURN
2540 VPOKE BASE(10)+INT(PY/8)*3
2+INT(PX/8)+32,255:RETURN
2550 VPOKE BASE(10)+INT(PY/8)*3
2+INT(PX/8)+32,165:RETURN
```

A linha 10 cuida do espaço *string* da memória (retire o comando **CLEAR** para ver o que acontece), das cores da tela e das teclas de função na parte inferior do vídeo. A linha 20 comunica que as cartas estão sendo embaralhadas.

As letras e números das cartas serão desenhados com **DRAW**. A linha 30 coloca as instruções para o desenho dentro da matriz **NU$**, depois que as leu na linha **DATA** 40. A linha 30 também dimensiona essa matriz.

Como não podemos usar sprites, colocaremos os símbolos dos naipes (os caracteres 3 a 6) diretamente na tela de alta resolução, com **VPOKE**. A linha 50 buscará os desenhos desses símbolos dentro da tabela de padrões da tela de 40 colunas. O endereço inicial da tabela de padrões da tela 0 começa em **BASE (2)**.

A linha 60 dimensiona a matriz **SQ**,

usada para embaralhar as cartas, e, ainda, seleciona a tela de alta resolução e inicializa o gerador de números randômicos.

O laço **FOR...NEXT** das linhas 120 a 140 transfere os padrões dos símbolos dos naipes para a tabela de padrões da tela de alta resolução. Além disso, coloca as cores desses símbolos na tabela de cores.

Como a tela de alta resolução é dividida em três porções, tudo deve ser feito três vezes. Assim, as linhas 130, 132 e 134 cuidam dos padrões, enquanto as linhas 131, 133 e 135 se encarregam das cores.

O laço **FOR...NEXT** das linhas 70 a 110 modifica a tabela de nomes da tela de alta resolução, impedindo que os símbolos sejam desenhados agora. Apague essas linhas para ver o que acontece.

A linha 160 coloca valores de 0 a 51 nos primeiros elementos de **SQ**. A linha 180 chama a sub-rotina que embaralha as cartas (linhas 1500 e 1510). O valor 0 é colocado em **N**; isso significa que o primeiro elemento de **SQ** está sendo colocado em questão.

A linha 190 inicia um laço **FOR... NEXT** que usa as variáveis **CX** e **CY**, cuja função é posicionar os símbolos dos naipes nas cartas.

A linha 200 chama duas sub-rotinas. A primeira, da linha 1000, fornece o naipe — **ST** — e o valor — **NM** — da carta em questão. A segunda, da linha 2000 à 2140, calcula as posições onde os símbolos dos naipes devem ser desenhados naquela carta.

Esta última sub-rotina parece complicada mas, com a ajuda de um baralho, pode-se entender melhor seu funcionamento. Vários padrões se repetem na disposição dos símbolos dos naipes — as cartas de menor valor geralmente usam um só deles, enquanto as de maior valor requerem três ou mais. A sub-rotina verifica quais desses padrões são necessários para colocar os símbolos de determinado naipe em número e posição adequados.

Antes do cálculo das posições dos naipes na tela, os números ou letras correspondentes ao valor da carta são desenhados em cantos opostos da mesma pelas linhas 2030 e 2040, que usam a informação contida em **NU$**.

As linhas 2050 a 2090, com base no valor da carta e de **CX** e **CY**, calculam as posições onde os símbolos dos naipes serão colocados na tela. As coordenadas da posição estão em **PX** e **PY**.

Uma vez que **PX** e **PY** tenham sido calculados, a sub-rotina da linha 2500 é chamada. Ela calcula a posição da tabela de nomes — **BASE (10)** — em que devemos colocar os símbolos dos naipes, para que eles apareçam numa posição correspondente a **PX,PY**. Essa sub-rotina é responsável pelo desenho propriamente dito.

A linha 2000 desenha a parte branca da carta, mas não apaga os desenhos produzidos com **VPOKE**. Isto é feito pela linha 2010, que apaga qualquer símbolo de naipe que tenha restado de uma antiga carta.

A linha 210 provoca uma pausa antes que o programa volte para a linha 190. Esta desenha uma nova série de oito cartas.

Apresentamos a seguir o programa que mostra as cartas na tela do Apple.

Para que funcione no TK-2000, precisaremos fazer uma pequena alteração.

```
10  HOME :E = 35000: HIMEM: E:
HTAB 9: VTAB 12: PRINT "EMBARAL
HANDO AS CARTAS"
20 F =  INT (E / 256): POKE 232
,E - F * 256: POKE 233,F
30  FOR I = E TO E + 41 + 17 *
32
40  READ A: POKE I,A
50  NEXT
60  SCALE= 1: ROT= 0
70  DATA  20 ,0 ,42 ,0 ,74 ,0 ,
106 ,0 ,138 ,0 ,170 ,0 ,202 ,0
,234 ,0 ,10 ,1 ,42 ,1 ,74 ,1 ,1
06 ,1 ,138 ,1 ,170 ,1 ,202 ,1 ,
234 ,1 ,10 ,2 ,42 ,2 ,74 ,2 ,10
6 ,2 ,138 ,2
72  DATA   0 ,72 ,45 ,109 ,209
,251 ,219 ,23 ,77 ,73 ,169 ,251
 ,219 ,27 ,110 ,73 ,9 ,213 ,63
,63 ,63 ,55 ,77 ,73 ,169 ,251 ,
219 ,27 ,6 ,0 ,0 ,0
74  DATA   0 ,72 ,45 ,109 ,209
,27 ,223 ,155 ,73 ,9 ,77 ,218 ,
59 ,63 ,159 ,9 ,77 ,73 ,218 ,21
9 ,251 ,74 ,45 ,109 ,209 ,219 ,
219 ,19 ,0 ,0 ,0 ,0
76  DATA   0 ,72 ,45 ,109 ,209
,251 ,219 ,83 ,73 ,9 ,141 ,219
,63 ,255 ,74 ,73 ,105 ,218 ,223
 ,219 ,74 ,45 ,109 ,209 ,219 ,2
19 ,19 ,0 ,0 ,0 ,0
78  DATA   0 ,72 ,77 ,77 ,218
,251 ,251 ,74 ,77 ,77 ,218 ,63
,63 ,159 ,73 ,9 ,77 ,218 ,251 ,
219 ,74 ,73 ,77 ,218 ,219 ,219
,2 ,0 ,0 ,0 ,0 ,0
80  DATA   0 ,72 ,45 ,109 ,209
,219 ,27 ,159 ,9 ,77 ,73 ,218 ,
59 ,63 ,159 ,73 ,9 ,77 ,218 ,25
1 ,219 ,74 ,45 ,109 ,209 ,219 ,
219 ,19 ,0 ,0 ,0 ,0
82  DATA   0 ,72 ,45 ,109 ,209
,219 ,27 ,159 ,9 ,77 ,73 ,218 ,
59 ,63 ,159 ,9 ,77 ,77 ,218 ,25
1 ,251 ,74 ,45 ,109 ,209 ,219 ,
219 ,19 ,0 ,0 ,0 ,0
84  DATA   0 ,72 ,45 ,45 ,141 ,
27 ,223 ,219 ,74 ,73 ,105 ,218
,251 ,219 ,74 ,9 ,77 ,209 ,219
,223 ,63 ,73 ,77 ,209 ,219 ,219
 ,19 ,0 ,0 ,0 ,0
86  DATA   0 ,72 ,45 ,109 ,209
,27 ,223 ,159 ,9 ,77 ,77 ,218 ,
59 ,63 ,159 ,9 ,77 ,77 ,218 ,25
1 ,251 ,74 ,45 ,109 ,209 ,219 ,
219 ,19 ,0 ,0 ,0 ,0
88  DATA   0 ,72 ,45 ,109 ,209
,27 ,223 ,159 ,9 ,77 ,77 ,218 ,
59 ,63 ,159 ,73 ,9 ,77 ,218 ,25
1 ,219 ,74 ,73 ,77 ,218 ,219 ,2
19 ,2 ,0 ,0 ,0 ,0
90  DATA   0 ,72 ,77 ,45 ,173 ,2
51 ,251 ,187 ,105 ,105 ,169 ,25
1 ,251 ,187 ,105 ,105 ,169 ,251
 ,251 ,187 ,105 ,41 ,45 ,213 ,2
19 ,219 ,19 ,0 ,0 ,0 ,0 ,0
92  DATA   0 ,72 ,9 ,45 ,141 ,2
19 ,223 ,155 ,73 ,9 ,77 ,218 ,2
51 ,219 ,74 ,73 ,77 ,218 ,251 ,
27 ,87 ,77 ,105 ,209 ,219 ,63 ,
159 ,0 ,0 ,0 ,0 ,0
94  DATA   0 ,8 ,45 ,45 ,109 ,2
18 ,223 ,27 ,87 ,77 ,9 ,141 ,27
 ,223 ,27 ,87 ,109 ,9 ,141 ,27
,223 ,31 ,87 ,45 ,45 ,109 ,218
,251 ,219 ,2 ,0 ,0
96  DATA   0 ,8 ,77 ,73 ,209 ,2
23 ,219 ,87 ,77 ,9 ,141 ,219 ,2
23 ,187 ,41 ,45 ,77 ,209 ,27 ,2
23 ,187 ,105 ,73 ,141 ,251 ,219
 ,187 ,0 ,0 ,0 ,0 ,0
98  DATA   0 ,72 ,109 ,109 ,26
,63 ,63 ,63 ,87 ,45 ,45 ,45 ,21
3 ,59 ,63 ,255 ,74 ,41 ,109 ,20
9 ,219 ,223 ,83 ,73 ,73 ,209 ,2
19 ,219 ,19 ,0 ,0 ,0
100  DATA   0 ,72 ,9 ,77 ,209 ,
27 ,63 ,223 ,74 ,45 ,45 ,141 ,5
9 ,63 ,63 ,191 ,9 ,45 ,45 ,141
,219 ,63 ,223 ,74 ,9 ,77 ,209 ,
219 ,219 ,19 ,0 ,0
102  DATA   0 ,72 ,41 ,109 ,209
 ,27 ,63 ,223 ,74 ,9 ,77 ,209 ,
255 ,31 ,191 ,41 ,45 ,45 ,173 ,
59 ,31 ,31 ,191 ,73 ,105 ,137 ,
219 ,63 ,223 ,2 ,0 ,0
104  DATA   0 ,72 ,9 ,77 ,209 ,
27 ,63 ,223 ,74 ,45 ,45 ,141 ,5
9 ,63 ,63 ,191 ,41 ,45 ,45 ,173
 ,27 ,31 ,31 ,159 ,73 ,105 ,137
 ,219 ,63 ,223 ,2 ,0
110  DIM SQ(61)
120  FOR K = 0 TO 51:SQ(K) = K:
 NEXT
160  HGR : POKE  - 16302,0
170  HCOLOR= 1: FOR I = 0 TO 19
1: HPLOT 0,I TO 279,I: NEXT
180  GOSUB 1500:N = 0
190  FOR CX = 6 TO 250 STEP 54:
 FOR CY = 2 TO 120 STEP 100
200  GOSUB 1000: GOSUB 2000: FO
R J = 1 TO 500: NEXT J,CY,CX
210  FOR J = 1 TO 1000: NEXT :
GOTO 190
1000 ST =  INT (SQ(N) / 13) + 1
:NM = SQ(N) - 13 * ST + 14:N =
N + 1: IF N > 51 THEN N = 0
1010  RETURN
1500  FOR X = 52 TO 2 STEP  - 1
1510 Q =  INT ( RND (1) * X) +
1
1520 T = SQ(X - 1):SQ(X - 1) =
SQ(Q - 1):SQ(Q - 1) = T
1530  NEXT
1540  FOR X = 0 TO 9:SQ(X + 52)
 = SQ(X): NEXT
1550  RETURN
2000  HCOLOR= 3: FOR I = CY TO
CY + 80: HPLOT CX,I TO CX + 50,
I: NEXT
2010 S = NM
2020  HCOLOR= 0
2030  DRAW S AT CX + 3,CY + 8
2040  ROT= 32: DRAW S AT CX + 4
8,CY + 72: ROT= 0
2050  IF NM / 2 <  > INT (NM /
2) OR NM > 10 THEN PX = CX + 2
8:PY = CY + 39: GOSUB 2500
2060  IF NM = 2 OR NM = 3 OR NM
 = 10 OR NM = 8 THEN PX = CX +
20:PY = CY + 27: GOSUB 2500:PY
= PY + 26:PX = PX + 8: ROT= 32:
 GOSUB 2500: ROT= 0
2080  IF NM < 4 OR NM > 10 THEN
 2140
2090  IF (NM = 10 OR NM = 8) TH
EN NS =  INT ((NM - 1) / 2): GO
TO 2100
2095 NS =  INT (NM / 2)
2100  FOR J = 0 TO NS - 1
2110 PX = CX + 8:PY = CY + 20 +
 J * 38 / (NS - 1):F = (J * 38
/ (NS - 1) + 20 > 39): ROT= F *
 32:PX = PX + F * 8:PY = PY + F
 * 2: GOSUB 2500: ROT= 0
2120 FE = (J * 38 / (NS - 1) +
20 < 34):F =  NOT (FE OR F):PX
= CX + 32 + ( NOT FE) * 8:PY =
PY + F * 2: ROT=  NOT FE * 32:
GOSUB 2500: ROT= 0
2130  NEXT
2140  RETURN
2500  ON ST GOTO 2510,2520,2530
,2540
2510  HCOLOR= 0: DRAW 14 AT PX,
PY: RETURN
2520  HCOLOR= 0: DRAW 15 AT PX,
PY: RETURN
2530  HCOLOR= 0: DRAW 16 AT  IN
T (PX), INT (PY): RETURN
2540  HCOLOR= 0: DRAW 17 AT  IN
T (PX), INT (PY): RETURN
```

Se você tiver um monitor monocromático e quiser dar um aspecto mais agradável à tela, apague a linha 170.

Os usuários do TK-2000 devem mudar a linha 160 para:

```
1160 MP : HGR
```

As primeiras linhas do programa são responsáveis por montar na memória do Apple uma tabela de figuras contendo as letras e números das cartas, bem como os símbolos dos naipes. As linhas **DATA** 70 a 104 foram criadas pelo nosso programa editor de figuras. As primeiras delas, com exceção da mensagem "EMBARALHANDO AS CARTAS", já apareceram diversas vezes em *INPUT*.

A linha 110 dimensiona a matriz **SQ**, usada para embaralhar as cartas. A linha 120 enche essa mesma matriz com valores de 0 a 51.

A linha 160 liga a tela de alta resolução. No Apple, um **POKE** faz com que a tela seja totalmente utilizada. No TK-2000, a tela selecionada é a segunda, e o comando **MP** precede **HGR**, para produzir o mesmo efeito.

A linha 170 desenha o pano de fundo da mesa de jogo.

A linha 180 chama a sub-rotina que embaralha as cartas e coloca o valor zero em N, o que significa que a primeira carta em SQ está em questão.

A linha 190 inicia dois laços **FOR...NEXT** que utilizam as variáveis **CX** e **CY**, usadas para posicionar os símbolos nas cartas.

A linha 200 chama duas sub-rotinas. A primeira, da linha 1000, fornece o naipe — **ST** — e o valor **NM** — da carta em questão. A segunda, da linha 2000 à 2140, calcula as posições em que devem ser desenhados os símbolos do naipe da mesma carta.

Esta última sub-rotina parece complicada mas, com a ajuda de um baralho, pode-se entender melhor seu funcionamento. Vários padrões se repetem na disposição dos símbolos dos naipes — as cartas de menor valor geralmente usam um só deles, enquanto as de maior valor requerem dois ou mais. A sub-rotina verifica quais desses padrões são necessários para desenhar os símbolos de determinado naipe em número e posição adequados. A instrução **ROT** complica um pouco mais essa sub-rotina, pois faz com que alguns símbolos apareçam de cabeça para baixo.

Antes do cálculo das posições, as linhas 2030 e 2040 desenham os números ou letras da carta em questão em cantos opostos da mesma, usando **DRAW**.

As linhas 2050 e 2090 calculam as posições de cada símbolo do naipe, com base no valor da carta. **PX** e **PY** são as coordenadas desta posição.

Uma vez que **PX** e **PY** tenham sido calculadas, a sub-rotina da linha 2500 é chamada. Essa rotina usa **DRAW** para desenhar os símbolos dos naipes.

A linha 210 provoca uma pausa antes que o programa volte para a linha 190. Esta desenha uma nova série de dez cartas.

T

Apresentamos a seguir a seção do programa para o TRS-Color que desenha as cartas. Digite-a e use **RUN** para ver o computador distribuir as cartas.

```
10 PMODE 3,1
20 CLS:PRINT @226,"ESTOU EMBARA
LHANDO AS CARTAS"
30 DIM NU$(13):FOR K=1 TO 13:RE
AD NU$(K):NEXT
40 DATA BD2S4U5ER5FD2NL5D3,RDLD
R,RDNLDL,DRDU2,NRDBGL,D2RNL,RS8
DGD,ND2RDNLDL,NDRBNLD,D2BL6BRS1
2NU2RU2L,S4BD5F2R2NL3D1,B4NR5D
6R3NH2NFR2U6,DND5NL4U2NEF
50 FOR K=1536 TO 1536+15*32 STE
P 32
60 READ A,B:POKE K,A:POKE K+1,B
:NEXT
70 DATA 60,224,184,184,238,236,
187,184,232,216,187,184,96,224,
159,176,14,192,12,128,2,0
80 DATA 2,0,3,0,1,128,14,192,5
6,176,238,236,59,176,14,192,12,
128,3,0,3,0
90 DATA 1,0,6,64,9,128,38,96,25
,144,102,100,153,152,102,100,15
3,152,34,32,1,0
100 DATA 2,0,9,128,6,64,9,128,3
4,32,153,152,102,101,153,152,10
2,101,17,16,2,0
110 DIM C(3),D(3),H(3),S(3),SQ(
61)
120 GET(0,0)-(13,10),H,G
130 GET(0,11)-(13,21),D,G
140 GET(0,22)-(13,32),S,G
150 GET(0,33)-(13,43),C,G
160 FOR K=0 TO 51:SQ(K)=K:NEXT
170 PCLS 6:SCREEN 1,1
180 GOSUB 1500:N=0
190 PCLS 6:FOR CX=6 TO 200 STEP
 50:FOR CY=11 TO 108 STEP 97
200 GOSUB 1000:GOSUB 2000:FOR J
=1 TO 500:NEXT J,CY,CX
210 FOR J=1 TO 1000:NEXT:GOTO 1
90
1000 ST=INT(SQ(N)/13)+1:NM=SQ(N
)-13*ST+14:N=N+1:IF N>51 THEN N
=0
1010 RETURN
1500 FOR X=52 TO 2 STEP -1
1510 Q=RND(X)
1520 T=SQ(X-1):SQ(X-1)=SQ(Q-1):
SQ(Q-1)=T
1530 NEXT
1540 FOR X=0 TO 9:SQ(X+52)=SQ(X
):NEXT
1550 RETURN
2000 LINE(CX,CY)-(CX+44,CY+72),
PRESET,BF
2010 S$=NU$(NM)
2020 IF ST>2 THEN COLOR 7 ELSE
COLOR 8
2030 DRAW"S12BM"+STR$(CX+3)+","
+STR$(CY+2)+S$
2040 DRAW"S12BM"+STR$(CX+35)+",
"+STR$(CY+64)+S$
2050 IF(NM/2 <> INT(NM/2) AND N
M<>7) OR NM>10 THEN PX=CX+16: P
Y=CY+31:GOSUB 2500
2060 IF NM=2 OR NM=3 OR NM=10 O
R NM=8 THEN PX=CX+16: PY=CY+19:
GOSUB 2500: PY=PY+24: GOSUB 250
0
2070 IF NM=7 THEN PX=CX+16: PY=
CY+39: GOSUB 2500
2080 IF NM<4 OR NM>10 THEN 2140
2090 IF (NM=10 OR NM=8) THEN NS
=INT((NM-1)/2) ELSE NS=INT(NM/2
)
2100 FOR J=0 TO NS-1
2110 PX=CX+3:PY=CY+12+J*38/(NS-
1):GOSUB 2500
2120 PX=CX+30:GOSUB 2500
2130 NEXT
2140 RETURN
2500 ON ST GOTO 2510,2520,2530,
2540
2510 PUT(PX,PY)-(PX+13,PY+10),H
,OR:RETURN
2520 PUT(PX,PY)-(PX+13,PY+10),D
,OR:RETURN
2530 PUT(PX,PY)-(PX+13,PY+10),C
,OR:RETURN
2540 PUT(PX,PY)-(PX+13,PY+10),S
,OR:RETURN
```

Escolhemos um **PMODE** com 4 cores, de modo que os naipes e números das cartas sejam azuis ou vermelhos. A linha 20 comunica ao jogador que as cartas estão sendo embaralhadas.

O comando **DRAW** coloca na tela de alta resolução as letras para os ases, reis, damas e valetes, bem como os números das cartas. As instruções para esse comando são lidas na linha **DATA** 40 e guardadas na matriz **NU$** pela linha 30, usando **READ**.

A linha 60 coloca os símbolos dos naipes na tela, usando **POKE**. Os padrões para esses símbolos são obtidos nas linhas **DATA** 70 a 90. As linhas 120 a 150 usam o comando **GET** para guardar os símbolos na memória, tão logo sejam desenhados. As matrizes que os guardam foram dimensionadas na linha 110, juntamente com a matriz **SQ**, empregada para embaralhar as cartas. A linha 160 coloca os números 0 a 51 na matriz **SQ**.

A linha 180 chama a sub-rotina que começa na linha 1500, para que as cartas sejam embaralhadas. O valor zero é colocado em **N**, o que significa que o elemento 0 da matriz **SQ** está em questão.

A linha 190 colore a tela de ciano e inicia dois laços **FOR...NEXT** que incluem as variáveis **CX** e **CY**, usadas posteriormente para posicionar os símbolos dos naipes nas cartas.

A linha 200 chama duas sub-rotinas. A primeira, da linha 1000, calcula o naipe — **ST** — e o número — **NM** — da carta em questão. A segunda, da linha 2000, calcula as posições em que os símbolos do naipe da carta serão desenhados.

Esta última sub-rotina parece complicada mas, com a ajuda de um baralho, pode-se entender melhor seu funcionamento. Existem certos padrões que se repetem na disposição dos símbolos de naipe — as cartas de menor valor usam apenas um deles, enquanto as de maior valor requerem dois ou mais. A sub-rotina verifica quais os padrões necessários para desenhar a carta em questão.

Antes do cálculo das posições do naipe, a letra ou o número da carta é desenhado em dois cantos opostos da mesma, com o comando **DRAW** e a informação contida em **NU$**.

As linhas 2050 a 2090 calculam as posições dos símbolos do naipe com base no valor da carta. **PX** e **PY** são as coordenadas dessas posições, calculadas a partir de **CX** e **CY**.

Uma vez que **PX** e **PY** tenham sido calculadas, a sub-rotina 2500 é chamada, desenhando os símbolos do naipe.

A linha 210 provoca uma pausa antes que o programa volte à linha 190. Esta apaga a tela e mostra outra carta.

FUNÇÕES MATEMÁTICAS

Aqui estão mais algumas funções matemáticas em BASIC. Suas aplicações práticas vão desde o cálculo da área de um piso até o cálculo da velocidade de um corpo em queda.

A função de potenciação tem variadas aplicações, sobretudo quando precisamos calcular áreas e volumes em nossos programas.

No computador, essa função, representada por ↑, é colocada sempre entre dois números. Por exemplo: 2↑3 (diz-se "dois elevado à terceira potência").

Na maioria dos computadores aqui indicados, a função de potenciação deve ser digitada por meio da tecla ^ (circunflexo), e é assim que ela aparecerá nas listagens. As exceções são:

Usa-se a tecla ↑ para digitar a operação de potenciação.

Usa-se a tecla ** (duplo asterisco) para digitar a operação de potenciação.

Um número elevado à potência de outro é simplesmente o primeiro número multiplicado por ele mesmo um certo número de vezes. O segundo número determina quantas vezes.

De volta ao exemplo inicial, dois elevado à terceira potência é dois multiplicado por dois três vezes. Ou seja:

```
2^3 = 2*2*2 = 8
```

A operação é a mesma para qualquer par de números.

```
2^5 = 2*2*2*2*2 = 32
5^4 = 5*5*5*5 = 625
```

Convencionalmente, representa-se a função de potenciação colocando-se o segundo número, em tamanho menor, junto ao primeiro. Por exemplo, 2^5, 5^4. O computador, porém, não entende essa notação matemática.

AO QUADRADO E AO CUBO

A potência de dois e a potência de três recebem denominações especiais. Qualquer número elevado à potência de dois é dito "elevado ao quadrado", e qualquer número elevado à potência de três é dito "elevado ao cubo". De fato, existem razões para esses nomes.

Quando calculamos a área de um retângulo (seja ele um tapete ou qualquer objeto retangular), medimos seu comprimento e largura e, em seguida, multiplicamos os dois números. O resultado dessa multiplicação é um número em unidades quadradas. A área é medida em "metros quadrados" porque, na verdade, estamos calculando quantos quadrados, de lados iguais a um metro, cabem nesta área. Da mesma maneira, um número elevado a dois é dito ao quadrado (uma área quadrada é representada pela multiplicação da unidade básica por ela mesma).

Enquanto a potência de dois recebe um nome de medida de área, a potência de três recebe um nome de medida de volume. Para calcular o volume de um cubo, multiplicamos o comprimento pela largura e, depois, pela altura. São necessárias, portanto, três multiplicações da unidade básica (duas para a área e mais uma para a altura). Um número elevado à terceira potência é, assim, um número ao cubo.

POTÊNCIAS MAIORES

Vimos que um número ao quadrado calcula área e um número ao cubo calcula volume. Não é possível construir modelos para potências maiores que três. Se tivéssemos um objeto com quatro dimensões, seu volume seria medido em unidades de quarta potência, o que é obviamente absurdo. Contudo, as potências maiores têm muita utilidade, como veremos a seguir.

Suponha, por exemplo, que queremos saber a probabilidade de um dado cair com a face seis para cima. É simples: cada face tem a mesma chance de aparecer; portanto, cada face tem 1/6 de chance. Suponha, agora, que queremos saber a probabilidade de obter seis duas vezes seguidas. Existe 1/6 de chance de que apareça um seis na primeira vez e, *se* aparecer, existe outro 1/6 de chance de que apareça um seis na segunda vez. A probabilidade total é 1/6 vezes 1/6, ou seja, 1/6 ao quadrado. O mesmo vale para quantas vezes quisermos. A chance de que o dado caia sete vezes com o três para cima, por exemplo, é: 1/6 * 1/6 * 1/6 * 1/6 * 1/6 * 1/6 * 1/6 = (1/6)↑7.

Quando examinamos a conversão para binário, tivemos a oportunidade de usar números elevados a potências maiores. Cada dígito binário representa uma potência do número dois. No número binário 11111111, por exemplo, o primeiro dígito da direita (bit 0) represen-

- A FUNÇÃO DE POTENCIAÇÃO
- NÚMEROS AO QUADRADO E NÚMEROS AO CUBO
- NÚMEROS ELEVADOS A GRANDES POTÊNCIAS
- O CÁLCULO DA ÁREA DE UM QUADRADO
- A RAIZ QUADRADA
- O USO DA FUNÇÃO **SQR** NUM PROGRAMA

ta um 1, o segundo (bit 1) um 2, o terceiro (bit 2) um 4 e os seguintes, respectivamente, 8, 16, 32, 64, 128.

Veja que 4 é 2*2, ou 2 elevado à potência de 2; 8 é 2*2*2, ou 2 elevado à terceira potência, 16 é 2*2*2*2 e assim por diante:

```
2^2 = 2*2             =  4
2^3 = 2*2*2           =  8
2^4 = 2*2*2*2         =  16
2^5 = 2*2*2*2*2       =  32
2^6 = 2*2*2*2*2*2     =  64
2^7 = 2*2*2*2*2*2*2   =  128
```

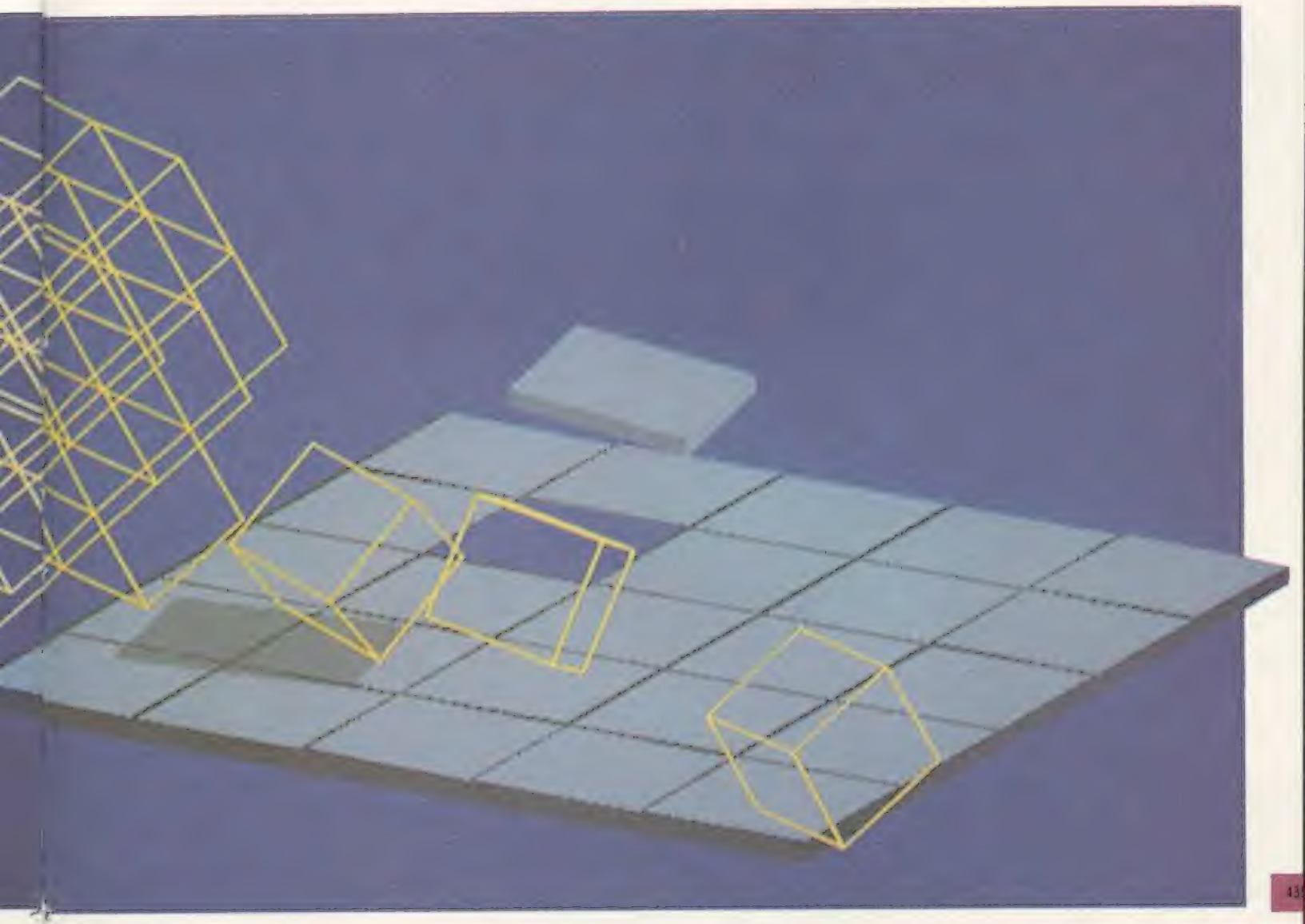
Estão faltando dois valores nessa tabela. Pode-se perceber facilmente quais são eles, mas a explicação talvez seja um tanto surpreendente. O primeiro valor é 2↑1. Obviamente, pela analogia binária, ele deve ser 2. Na verdade, se 2↑2 é 2 multiplicado por ele mesmo uma só vez, 2↑1 deve ser 2 multiplicado por ele mesmo uma vez a menos, ou seja, nenhuma vez. 2↑1 fica, então, valendo 2.

O valor abaixo deste é 2↑0. À primeira vista, parece que o resultado é zero mas, pela tabela acima, fica claro que é 1.

Isso acontece porque 2 precisa ser multiplicado por ele mesmo uma vez a menos que 2↑1. Como 2↑1 é 2 mesmo, só há uma maneira de multiplicar uma vez a menos: dividir 2 por ele mesmo. E qualquer número dividido por ele mesmo, como sabemos, vale 1.

Para verificar o que foi dito ou o valor de qualquer outra potência, você deve digitar:

```
PRINT 2^X
```



...onde X é o valor que pretende verificar. Em seguida, tecle <**ENTER**> ou <**RETURN**>.

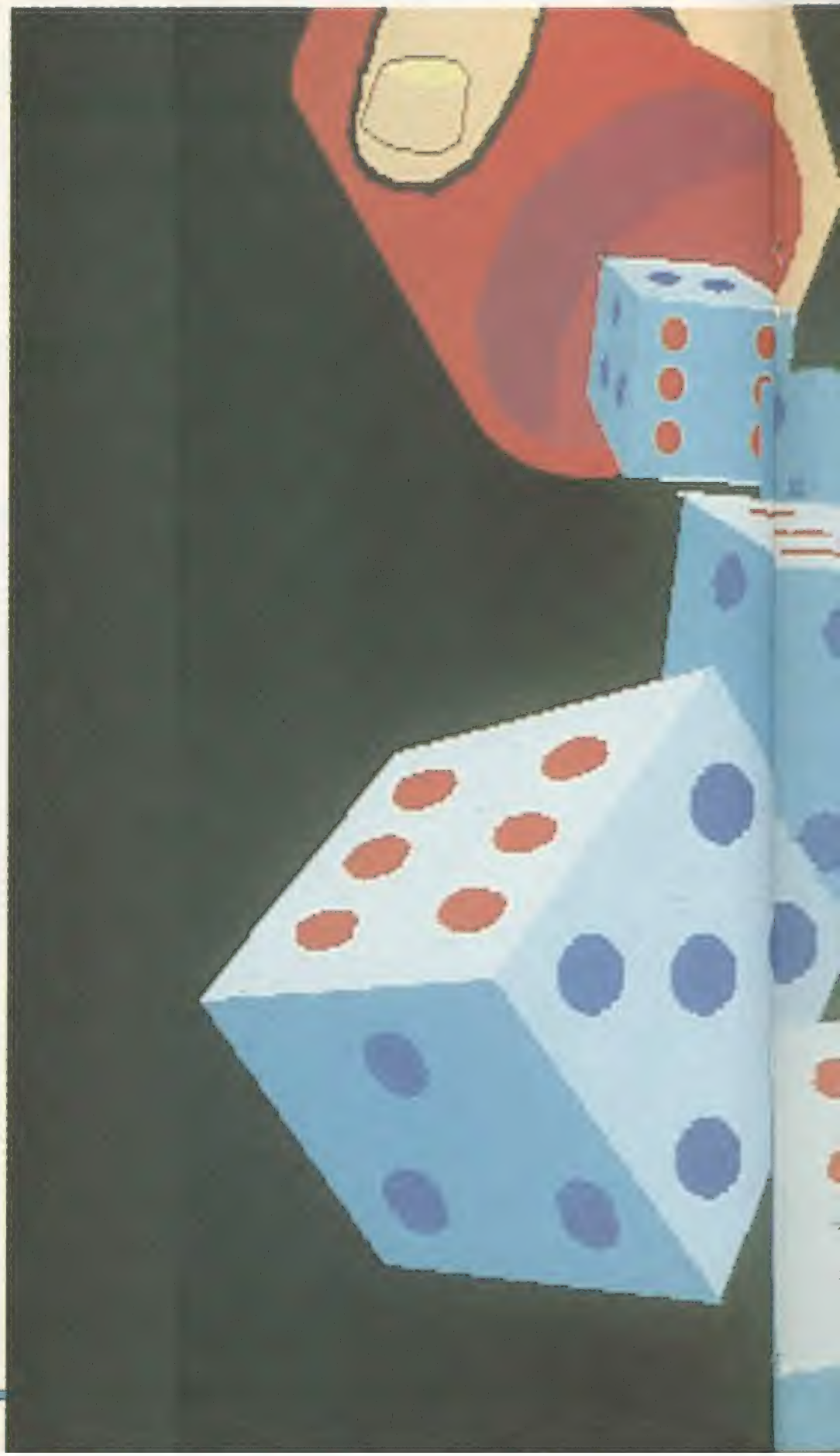
POR CURIOSIDADE...

Experimente atribuir qualquer valor a X, ainda que valores grandes possam acarretar erros de sobrecarga (overflow). O que aconteceria, por exemplo, se se fizesse X igual a 1/2? Retornaremos ao assunto mais tarde. Por enquanto, vamos explorar um pouco mais a tão usada potência quadrada.

COMPARE QUADRADOS

O programa abaixo usa uma série de números para desenhar na tela diferentes quadrados, permite que se escolha dois deles e compara suas áreas. Acompanhando-o, a idéia de potência ficará mais clara.

```
10 SCREEN 2:COLOR 15,12,3
20 FOR N=17 TO 1 STEP -1
30 LINE(60,171)-(60+8*N,171-8*N
),1,B
40 NEXT
50 IF INKEY$="" THEN 50
60 CLS:SCREEN 0
70 PRINT"Deseja comparar alguma
 área (s/n) ?"
80 A$=INKEY$:IF A$<>"s" AND A$<
>"n" AND A$<>"S" AND A$<>"N" TH
EN 80
90 IF A$="n" OR A$="N" THEN 270
100 PRINT:INPUT"Qual o primeiro
 quadrado cuja área quer compar
ar (1-17) ";A:A=INT(A):IF A<1 O
R A>17 THEN 100
110 PRINT:INPUT"E qual é o segu
ndo (1-17) ";B:B=INT(B):IF B<1
OR B>17 THEN 110
120 IF A>B THEN 220
130 CLS:PRINT" A primeira área
cabe ";
140 PRINTUSING"###.##";B^2/A^2;
150 PRINT" vezes den-tro da seg
unda"
160 FOR K=1 TO 2500:NEXT
170 CLS:SCREEN 2
180 LINE(60,171)-(60+8*A,171-8*
A),1,BF:LINE(60,171)-(60+8*B,17
1-8*B),1,B
190 IF INKEY$="" THEN 190
200 CLS:SCREEN 0:PRINT" Quer co
mparar mais alguma área (s/n) ?
"
```



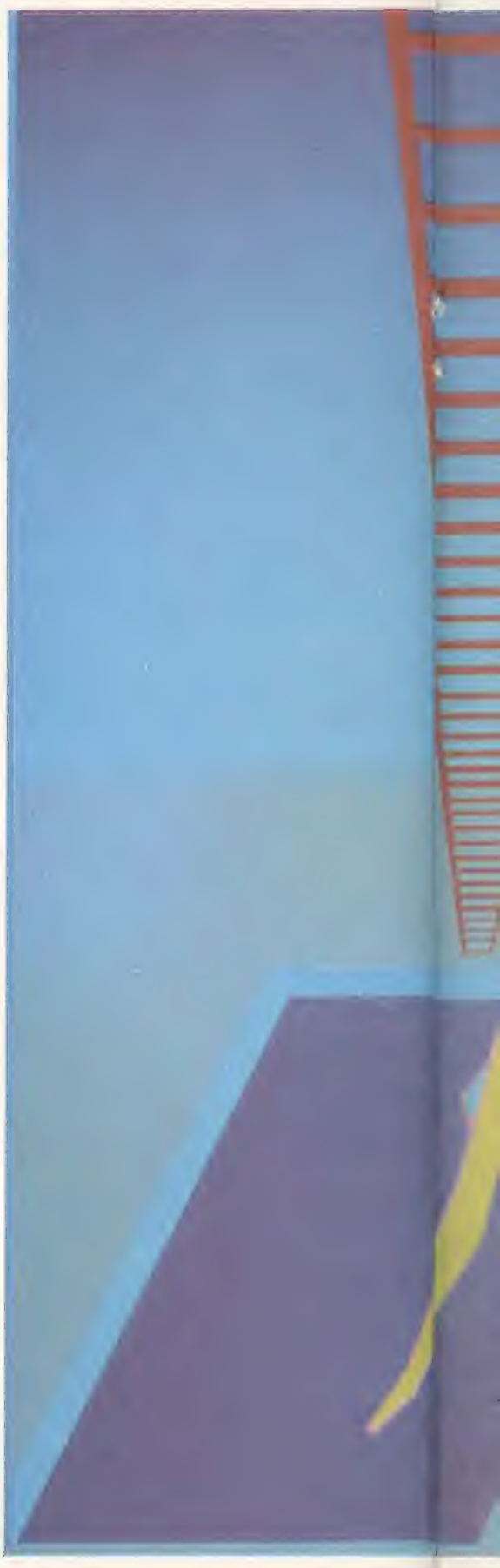
```
210 GOTO 80
220 CLS:PRINT" A primeira área
é ";
230 PRINTUSING"###.##";A^2/B^2;
240 PRINT" vezes maior que a se
gunda"
250 SWAP A,B
260 GOTO 160
270 CLS
280 LOCATE 5,10
290 PRINT"qualquer tecla para r
einiciar"
300 IF INKEY$="" THEN 300
310 GOTO 10
```

```
10  HGR2 : HCOLOR= 3
20  HPLOT 10,180 TO 180,180
30  HPLOT 10,10 TO 10,180
40  FOR I = 10 TO 170 STEP 10
50  HPLOT 10,I TO 190 - I,I
60  HPLOT 190 - I,I TO 190 - I,
180
70  NEXT I
80  FOR T = 0 TO 3000: NEXT : T
EXT
```

```
90  HOME : VTAB 10
100  PRINT "DESEJA COMPARAR ALG
UMA AREA (S/N) ?"
110  GET A$: IF A$ <  > "S" AND
 A$ <  > "N" THEN 110
120  IF A$ = "N" THEN 500
130  PRINT : INPUT "QUAL O PRIM
EIRO QUADRADO QUE DESEJA COM-PA
RAR (1-17) ";A:A =  INT (A): IF
 A < 1 OR A > 17 THEN 130
140  PRINT : INPUT "E QUAL O SE
GUNDO (1-17) ";B:B =  INT (B):
IF B < 1 OR B > 17 THEN 140
150  IF A > B THEN 190
160  HOME : VTAB 10
170  PRINT "A PRIMEIRA AREA CAB
E ";B ^ 2 / A ^ 2;" VEZES DENTR
O DA SEGUNDA"
180  GOTO 210
190  HOME : VTAB 10
200  PRINT "A PRIMEIRA AREA E'
";A ^ 2 / B ^ 2;" VEZES MAIOR Q
UE A SEGUNDA"
210  FOR T = 0 TO 2000: NEXT
220  HGR2 : HCOLOR= 3
230 KA = (A + 1) * 10:LA = 190
- KA
240 KB = (B + 1) * 10:LB = 190
- KB
250  HPLOT 10,LA TO KA,LA TO KA
,180 TO 10,180 TO 10,LA
260  HPLOT 10,LB TO KB,LB TO KB
,180 TO 10,180 TO 10,LB
270  FOR T = 0 TO 3000: NEXT
500  TEXT : HOME : VTAB 10
510  PRINT "QUALQUER TECLA PARA
 REINICIAR";
520  GET A$: IF A$ = "" THEN 52
0
530  RUN
```

```
 10 CLS
 20 PRINT AT 2,0;"Comprimento"
 ;AT 3,0;"dos";AT 4,0;"lados"
 30 PRINT AT 2,24;"Area";AT 3,
 24;"do";AT 4,24;"quadrado"
 40 LET z=0: PLOT 55,23
 50 FOR n=1 TO 13
 60 LET m=n: IF m>7 THEN  LET
 m=m-8
 70 DRAW 0,n*8
 80 DRAW  PAPER m;n*8,0
 90 DRAW  PAPER m;0,-(n*8)
100 DRAW  PAPER 7;-(n*8),0
110 PRINT AT 6+(13-n),0;n;AT 6
+(13-n),25;n*n
120 PAUSE 10: NEXT n
140 INPUT "Voce quer comparar
areas (s/n)?";a$
150 IF a$<>"s" AND a$<>"n"
THEN  GOTO 140
160 IF a$="n" THEN  GOTO 300
170 INPUT "Qual e o primeiro q
uadrado cuja area voce quer co
mparar? (1-13)";a: IF a<1 OR a
>13 THEN  GOTO 170
180 INPUT "E qual e o segundo?
(1-13)";b: IF b<1 OR b>13 THEN
 GOTO 180
190 LET x=INT (a): LET y=INT (
b)
200 IF x>y THEN  GOTO 280
210 CLS : PRINT "A segunda are
a e ";y^2/x^2;" vezes maior do
 que a primeira"
220 PLOT 20,0: DRAW x*8,0:
DRAW 0,x*8: DRAW -x*8,0: DRAW
0,-x*8: DRAW y*8,0: DRAW 0,y*8
: DRAW -y*8,0: DRAW 0,-y*8
230 PRINT "Voce quer comparar
mais areas   (s/n)?"
240 INPUT a$
250 IF a$<>"s" AND a$<>"n"
THEN  GOTO 240
260 IF a$="n" THEN  LET z=1:
GOTO 300
270 GOTO 170
280 CLS : PRINT "A primeira ar
ea e ";x^2/y^2;" vezes maior d
o que a segunda"
290 GOTO 220
300 IF z=1 THEN  CLS : LET z=0
310 PRINT  INK 2; FLASH 1;AT 0
,0;" Pressione qualquer tecla
para   recomecar"
320 PAUSE 0
330 IF INKEY$="n" THEN  STOP
340 RUN
```

```
10 PMODE4,1:PCLS:SCREEN1,1
20 FOR N=17 TO 1 STEP -1
30 LINE (20,171)-(20+8*N,171-8*
N),PSET,B
40 NEXT
50 CLS
60 IF INKEY$="" THEN 60
70 PRINT" VOCE QUER COMPARAR AR
EAS (S/N) ?"
80 A$=INKEY$:IF A$<>"S" AND A$<
>"N" THEN 80
90 IF A$="N" THEN 220
100 PRINT:INPUT" QUAL E O PRIME
IRO QUADRADO CUJA AREA VOCE QUE
R COMPARAR (1-17)";A:A=INT(A):I
F A<1 OR A>17 THEN 100
110 PRINT:INPUT" E QUAL E O SEG
UNDO (1-17)";B:B=INT(B):IF B<1
OR B>17 THEN 110
120 IF A>B THEN 200
130 CLS:PRINT" A SEGUNDA AREA E
 ";B^2/A^2;"VEZES    MAIOR DO Q
UE A PRIMEIRA"
140 FOR K=1 TO 6000:NEXT
150 PCLS:SCREEN 1,1
160 LINE(20,171)-(20+8*A,171-8*
A),PSET,B:LINE(20,171)-(20+8*B,
171-8*B),PSET,B
170 IF INKEY$="" THEN 170
180 CLS:PRINT" VOCE QUER COMPAR
AR MAIS AREAS (S/N) ?"
190 GOTO 80
200 CLS:PRINT" A PRIMEIRA AREA
E";A^2/B^2:PRINT @32," VEZES MA
IOR DO QUE A SEGUNDA."
210 GOTO 140
220 CLS
230 PRINT @33,"PRESSIONE QUALQU
ER TECLA PARA   RECOMECAR"
240 IF INKEY$="" THEN 240
250 GOTO 10
```



A rotina que faz a comparação começa pedindo que se escolha o primeiro dos dois quadrados que serão cotejados. Em seguida, verifica se o número que entrou não é ilegal, ou melhor, se não é menor ou maior que o número de quadrados.

A função **INT** transforma em inteiro o número que entrou, caso este seja um decimal.

Uma vez escolhidos os dois números, o computador compara-os a fim de determinar qual é o maior. Em seguida, eleva cada número ao quadrado (isto é, multiplica cada um por si mesmo) e divide o maior pelo menor.

Finalmente, obtido o resultado, a rotina joga na tela a mensagem que diz qual número é o maior e o quanto ele é maior que o outro.

RAIZ QUADRADA

A potência de dois, ou a função quadrada, talvez seja a mais utilizada das potências. Mas muitas vezes precisamos usá-la no modo inverso, ou seja, achar o número que gerou o número ao quadrado que conhecemos. Suponha que sabemos, por exemplo, que a área de um quadrado é 81 e queremos especificar o comprimento dos lados.

Com o número 81 não é tão difícil: 9 vezes 9 é 81; portanto, o comprimento de cada lado deve ser 9. Mas se a área fosse 127, por exemplo, o cálculo do comprimento do lado (ou do número que ao quadrado é 127) torna-se bem mais difícil. Os computadores possuem uma função que ajuda neste cálculo: a função raiz quadrada.

Digite **PRINT SQR(81)** e tecle <**ENTER**> ou <**RETURN**> — o número 9 deverá aparecer na tela. Para saber quanto vale a raiz quadrada de 127, basta repetir a operação, substituindo o 81 dentro dos parênteses por 127.

O computador dispõe do comando especial **SQR** porque a raiz quadrada é amplamente empregada. Mas podemos utilizar também, para o mesmo cálculo, a função de potenciação.

Se você já usou frações como potência, deve ter observado que 2↑.5 (ou dois elevado a 1/2) tem o mesmo efeito que **SQR(2)**. Experimente em seu micro **SQR(81)** e depois 81↑.5. Os resultados podem não ser exatamente os mesmos mas, certamente, serão bem próximos.

A fração 1/3, usada como potência, calcula o inverso do cubo, ou seja, a raiz cúbica. A mesma analogia vale para as outras potências. Embora cada raiz tenha sua própria aplicação, a raiz quadrada é a mais usada de todas.

O comando **SQR** faz muito mais do que simplesmente calcular o lado de um quadrado do qual se conhece a área. Algumas equações matemáticas têm números ao quadrado e raízes quadradas e, nesses casos, pode-se utilizar **SQR** para calcular o resultado no computador.

O programa que apresentamos a seguir usa uma equação para calcular o tempo de queda de um objeto (desprezando-se a resistência do ar). Os físicos chamam essa situação de "corpo em queda livre no vácuo". O programa também calcula a velocidade em que o objeto se encontra quando atinge o solo. Esses cálculos envolvem números ao quadrado e raízes quadradas porque qualquer objeto em queda e sob a influência da gravidade cai cada vez mais rápido com o passar do tempo (desprezando-se a resistência do ar). Na verdade, estamos tratando com o quadrado do tempo. Mas, antes de maiores detalhes, vejamos o programa funcionando.

```
10 CLS
20 PRINT"Qual a altura da queda
 ";:INPUT D
30 IF D<0 THEN 20
40 T=SQR((2*D)/9.81)
50 V=SQR(2*D*9.81)
60 T=INT(T*100)/100
70 V=INT(V*100)/100
80 PRINT:PRINT"Tempo até chegar
":PRINT"ao solo =";T;"segundos"
90 PRINT:PRINT"Velocidade máxim
a":PRINT"de impacto =";V;"metro
s por segundo"
100 PRINT"(";1.6*INT(2.25*V+.5)
;" Km/h)"
200 LOCATE 4,20:PRINT"Qualquer
tecla para reinicio"
210 IF INKEY$="" THEN 120
220 RUN
```

```
10  HOME
20  PRINT "QUAL A ALTURA DA QUE
DA ";: INPUT D
30  IF D < 0 THEN 20
40 T =  SQR ((2 * D) / 9.81)
50 V =  SQR (2 * D * 9.81)
60 T =  INT (T * 100) / 100
70 V =  INT (V * 100) / 100
80  PRINT : PRINT "TEMPO ATE CH
EGAR": PRINT "AO SOLO = ";T;" S
EGUNDOS"
90  PRINT : PRINT "VELOCIDADE M
AXIMA": PRINT "DE IMPACTO ";V;"
 METROS POR SEGUNDO"
100  PRINT "(";1.6 *  INT (2.25
 * V + .5);" KM/H)"
200  VTAB 20: PRINT "QUALQUER T
ECLA PARA REINICIO"
210  GET A$: IF A$ = "" THEN 21
0
220  RUN
```

```
 10 CLS
 20 INPUT "INTRODUZA A DISTANC
 IA DA QUEDA (METROS)",D
 30 IF D<0 THEN  GOTO 20
 40 LET T=SQR ((2*D)/9.81)
 50 LET V=SQR (2*D*9.81)
 60 LET T=INT (T*100)/100
 70 LET V=INT (V*100)/100
 80 PRINT  INVERSE 1''"TEMPO P
 ARA CHEGAR AO CHAO:"; INVERSE
  0'T;" SEGUNDOS"
 90 PRINT  INVERSE 1'"VELOCIDA
 DE MAXIMA ATINGIDA:"; INVERSE
  0'V;" METROS POR SEGUNDO"
100 PRINT "(";INT (2.25*V+.5);
" MPH)"
200 PRINT ''"PRESSIONE QUALQUE
R TECLA PARA   RECOMECAR"
210 IF INKEY$="" THEN  GOTO
 210
220 GOTO 10
```

```
10 CLS
20 INPUT "DIGITE A DISTANCIA DA
 QUEDA (EM METROS)";D
30 IF D<0 THEN 10
40 T=SQR((2*D)/9.81)
50 V=SQR(2*D*9.81)
60 T=INT(T*100)/100
70 V=INT(V*100)/100
80 PRINT @97,"tempo para chegar
 ao chao:":PRINT T;"SEGUNDOS"
90 PRINT @225,"velocidade maxim
a alcancada:":PRINT V;"METROS P
OR SEGUNDO"
100 PRINT " (";INT(2.25*V+.5);"
MPH)"
110 PRINT:PRINT"PRESSIONE QUALQ
UER TECLA PARA   RECOMECAR"
120 IF INKEY$="" THEN 120
130 GOTO 10
```

Em primeiro lugar, o computador limpa a tela e pergunta pela altura da qual o objeto vai cair. Caso você responda com um número negativo, a pergunta será feita novamente, pois não se pode calcular raiz quadrada de um número negativo.

Depois o computador calcula o tempo que o objeto levará para chegar ao chão e a velocidade com que chega. Isso é feito nas linhas 40 e 50.

A equação usada na linha 40 é uma das versões da "equação de movimento" que, depois de rearranjada, tomou a forma:

$$T = \sqrt{(2*D)/a}$$

...onde **T** é o tempo necessário para percorrer determinada distância (nesse caso, para atingir o solo), **D** é a distância (fornecida por você) e **a** é a aceleração (mede o quanto a velocidade aumenta em cada segundo).

No programa não há nenhuma variável **a**, já que seu valor é constante: no lugar de **a** vemos, assim, 9.81.

A linha 40 resolve a equação, indicando o tempo que o objeto leva para atingir o solo.

A linha seguinte resolve uma equação semelhante, que calcula a velocidade do objeto no momento em que atinge o solo.

$$V = \sqrt{2*D*9.81}$$

Nessa equação, em vez de dividir, o computador multiplica o número 2*D pela aceleração. V significa, no caso, velocidade.

Em ambas as equações, o sinal $\sqrt{}$ sobre o "2*D*9.81" ou "2*D39.81" significa raiz quadrada de — é aqui, portanto, que usamos a raiz quadrada. Uma vez obtida a resposta para uma das equações, pode-se mudá-la de modo que o computador calcule a altura da qual o objeto foi jogado.

De um modo geral, a equação se aplica a qualquer objeto em queda, seja ele um tijolo ou um foguete. O que pode mudar é o valor da aceleração. No nosso exemplo, a aceleração deve-se à gravidade; por isso, **a** é ajustado para este valor. A gravidade tem uma aceleração de 9.81 metros por segundo por segundo. Em outras palavras, para cada segundo na queda de um objeto sua velocidade aumenta de 9.81 metros por segundo. Metros por segundo por segundo também pode ser escrito como metros por segundo ao quadrado — e é aqui que a função potência entra em cena.

As equações para esse cálculo tomam a seguinte forma:

$$D = \frac{a*(t\hat{}2)}{2}$$

ou

```
D = V^2/(a*2)
```

Tente modificar o programa de modo que ele calcule respostas para estas equações. Em vez de altura, poderíamos fornecer ao computador a velocidade com que desejamos que um objeto atinja o solo (para a segunda equação) ou o tempo que ele deve levar antes do impacto. Em ambos os casos, o computador calcularia a altura necessária para satisfazer o dado fornecido.

A capacidade de resolução de problemas desse tipo tem vários usos práticos. Na maioria deles, porém, as equações necessitam de um tratamento mais cuidadoso, que leve em conta influências sobre o corpo em movimento. Alguns exemplos de aplicação seriam o cálculo do desempenho de um carro em testes de frenagem e aceleração ou mesmo a reconstituição de um acidente automobilístico. Neste último caso, poderíamos precisar a velocidade em que se encontrava o veículo no momento da batida. O cálculo é possível, contanto que conheçamos os valores corretos para satisfazer as equações.

Por meio da função de potenciação pode-se calcular muitas outras coisas além do desempenho de um carro ou da área de uma casa. Ela permite determinar, por exemplo, o quanto uma árvore cresce, ou por que um pássaro do tamanho de um elefante não pode levantar vôo. Num próximo artigo continuaremos a explorar o uso das funções matemáticas no computador, inclusive para a obtenção de efeitos gráficos.

Por que recebo uma mensagem de erro quando tento calcular a raiz quadrada de um número negativo?

Os computadores não são capazes de calcular a raiz quadrada de um número negativo e, na verdade, isto não existe no universo dos reais (embora para certos propósitos se atribua um valor imaginário).

Elevar um número ao quadrado significa multiplicá-lo por ele mesmo. Números positivos multiplicados entre si resultam num número positivo, mas o mesmo acontece com os negativos (negativo vezes negativo resulta num número positivo). Não existe nenhum número real que, elevado ao quadrado, resulte num número negativo. E é claro que o computador acusa erro quando lhe pedimos para fazer o impossível.

Quando usamos **SQR** num programa precisamos estar certos de que o número em questão será sempre positivo. Caso haja a possibilidade de que apareçam números negativos (resultados de cálculos anteriores, por exemplo), aconselha-se o uso da função **ABS**. Essa função converte um número em seu valor absoluto, ou seja, retira o sinal negativo, se houver. Para incluí-la num programa, substitua **SQR(A)** por **SQR(ABS(A))**, onde **A** representa o número que está usando.

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR LINHA

LINHA	FABRICANTE	MODELO
Apple II +	Appletronica	Thor 2010
Apple II +	CCE	MC-4000 Exato
Apple II +	CPA	Absolutus
Apple II +	CPA	Polaris
Apple II +	Digitus	DGT-AP
Apple II +	Dismac	D-8100
Apple II +	ENIAC	ENIAC II
Apple II +	Franklin	Franklin
Apple II +	Houston	Houston AP
Apple II +	Magnex	DM II
Apple II +	Maxitronica	MX-2001
Apple II +	Maxitronica	MX-48
Apple II +	Maxitronica	MX-64
Apple II +	Maxitronica	Maxitronic I
Apple II +	Microcraft	Craf II Plus
Apple II +	Milmar	Apple II Plus
Apple II +	Milmar	Apple Master
Apple II +	Milmar	Apple Senior
Apple II +	Omega	MC-400
Apple II +	Polymax	Maxxi
Apple II +	Polymax	Poly Plus
Apple II +	Spectrum	Microengenho I
Apple II +	Spectrum	Spectrum ed
Apple II +	Suporte	Venus II
Apple II +	Sycomig	SIC I
Apple II +	Unitron	AP II
Apple II +	Victor do Brasil	Elppa II Plus
Apple II +	Victor do Brasil	Elppa Jr.
Apple IIe	Microcraft	Craft IIe
Apple IIe	Microdigital	TK-3000 IIe
Apple IIe	Spectrum	Microengenho II
MSX	Gradiente	Expert GPC-1
MSX	Sharp	Hotbit HB-8000
Sinclair Spectrum	Microdigital	TK-90X
Sinclair Spectrum	Timex	Timex 2000
Sinclair ZX-81	Apply	Apply 300
Sinclair ZX-81	Engebras	AS-1000
Sinclair ZX-81	Filcres	NEZ-8000
Sinclair ZX-81	Microdigital	TK-82C
Sinclair ZX-81	Microdigital	TK-83
Sinclair ZX-81	Microdigital	TK-85
Sinclair ZX-81	Prologica	CP-200
Sinclair ZX-81	Ritas	Ringo R-470
Sinclair ZX-81	Timex	Timex 1000
Sinclair ZX-81	Timex	Timex 1500
TRS-80 Mod. I	Dismac	D-8000
TRS-80 Mod. I	Dismac	D-8001/2
TRS-80 Mod. I	LNW	LNW-80
TRS-80 Mod. I	Video Genie	Video Genie I
TRS-80 Mod.III	Digitus	DGT-100
TRS-80 Mod.III	Digitus	DGT-1000
TRS-80 Mod.III	Kemitron	Naja 800
TRS-80 Mod.III	Prologica	CP-300
TRS-80 Mod.III	Prologica	CP-500
TRS-80 Mod.III	Sysdata	Sysdata III
TRS-80 Mod.III	Sysdata	Sysdata Jr.
TRS-80 Mod.IV	Multix	MX-Compacto
TRS-80 Mod.IV	Sysdata	Sysdata IV
TRS-Color	Codimex	CS-6508
TRS-Color	Dynacom	MX-1600
TRS-Color	LZ	Color 64
TRS-Color	Microdigital	TKS-800
TRS-Color	Prologica	CP-400

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR NOME

FABRICANTE	MODELO	PAÍS	LINHA
Appletronica	Thor 2010	Brasil	Apple II +
Apply	Apply 300	Brasil	Sinclair ZX-81
CCE	MC-4000 Exato	Brasil	Apple II +
CPA	Absolutus	Brasil	Apple II +
CPA	Polaris	Brasil	Apple II +
Codimex	CS-6508	Brasil	TRS-Color
Digitus	DGT-100	Brasil	TRS-80 Mod.III
Digitus	DGT-1000	Brasil	TRS-80 Mod.III
Digitus	DGT-AP	Brasil	Apple II +
Dismac	D-8000	Brasil	TRS-80 Mod. I
Dismac	D-8001/2	Brasil	TRS-80 Mod. I
Dismac	D-8100	Brasil	Apple II +
Dynacom	MX-1600	Brasil	TRS-Color
ENIAC	ENIAC II	Brasil	Apple II +
Engebras	AS-1000	Brasil	Sinclair ZX-81
Filcres	NEZ-8000	Brasil	Sinclair ZX-81
Franklin	Franklin	USA	Apple II +
Gradiente	Expert GPC1	Brasil	MSX
Houston	Houston AP	Brasil	Apple II +
Kemitron	Naja 800	Brasil	TRS-80 Mod.III
LNW	LNW-80	USA	TRS-80 Mod. I
LZ	Color 64	Brasil	TRS-Color
Magnex	DM II	Brasil	Apple II +
Maxitronica	MX-2001	Brasil	Apple II +
Maxitronica	MX-48	Brasil	Apple II +
Maxitronica	MX-64	Brasil	Apple II +
Maxitronica	Maxitronic I	Brasil	Apple II +
Microcraft	Craft II Plus	Brasil	Apple II +
Microcraft	Caft IIe	Brasil	Apple IIe
Microdigital	TK-3000 IIe	Brasil	Apple IIe
Microdigital	TK-82C	Brasil	Sinclair ZX-81
Microdigital	TK-83	Brasil	Sinclair ZX-81
Microdigital	TK-85	Brasil	Sinclair ZX-81
Microdigital	TK-90X	Brasil	Sinclair Spectrum
Microdigital	TKS-800	Brasil	TRS-Color
Milmar	Apple II Plus	Brasil	Apple II +
Milmar	Apple Master	Brasil	Apple II +
Milmar	Apple Senior	Brasil	Apple II +
Multix	MX-Compacto	Brasil	TRS-80 Mod.IV
Omega	MC-400	Brasil	Apple II +
Polymax	Maxxi	Brasil	Apple II +
Polymax	Poly Plus	Brasil	Apple II +
Prologica	CP-200	Brasil	Sinclair ZX-81
Prologica	CP-300	Brasil	TRS-80 Mod.III
Prologica	CP-400	Brasil	TRS-Color
Prologica	CP-500	Brasil	TRS-80 Mod.III
Ritas	Ringo R-470	Brasil	Sinclair ZX-81
Sharp	Hotbit HB-8000	Brasil	MSX
Spectrum	Microengenho I	Brasil	Apple II +
Spectrum	Microengenho II	Brasil	Apple IIe
Spectrum	Spectrum ed	Brasil	Apple II +
Suporte	Venus II	Brasil	Apple II +
Sycomig	SIC I	Brasil	Apple II +
Sysdata	Sysdata III	Brasil	TRS-80 Mod.III
Sysdata	Sysdata IV	Brasil	TRS-80 Mod.IV
Sysdata	Sysdata Jr.	Brasil	TRS-80 Mod.III
Timex	Timex 1000	USA	Sinclair ZX-81
Timex	Timex 1500	USA	Sinclair ZX-81
Timex	Timex 2000	USA	Sinclair Spectrum
Unitron	AP II	Brasil	Apple II +
Victor do Brasil	Elppa II Plus	Brasil	Apple II +
Victor do Brasil	Elppa Jr.	Brasil	Apple II +
Video Genie	Video Genie I	USA	TRS-80 Mod. I

UM LOGOTIPO PARA CADA MODELO DE COMPUTADOR

INPUT foi especialmente projetado para microcomputadores compatíveis com as sete principais linhas existentes no mercado.
Os blocos de textos e listagens de programas aplicados apenas a determinadas linhas de micros podem ser identificados por meio dos seguintes símbolos:

Sinclair ZX-81

TRS-80

TK-2000

MSX

Spectrum

TRS-Color

Apple II

Quando o emblema for seguido de uma faixa, então tanto o texto como os programas que se seguem passam a ser específicos para a linha indicada.

NO PRÓXIMO NÚMERO

PROGRAMAÇÃO BASIC
Programas inteiramente corretos podem apresentar erros inesperados. Veja como evitá-los.

PERIFÉRICOS
Já existem sintetizadores de voz para a maioria dos micros domésticos. Conheça os principais tipos e seus efeitos.

PROGRAMAÇÃO DE JOGOS
Digite mais uma rotina para seu jogo de Vinte-e-um: ela lhe permitirá apostar e pedir mais cartas.

PROGRAMAÇÃO BASIC
Para evitar trabalho desnecessário, aprenda a combinar programas.